NEURAL SEMANTIC PARSING, ANNOTATION AND EVALUATION FOR TURKISH

TÜRKÇE İÇİN NÖRAL SEMANTİK AYRIŞTIRMA, ETİKETLEME VE DEĞERLENDİRME

NECVA BÖLÜCÜ

ASSOC. PROF. DR. HARUN ARTUNER
Supervisor
ASSOC. PROF. DR. BURCU CAN BUĞLALILAR
2nd Supervisor

Submitted to
Graduate School of Science and Engineering of Hacettepe University
as a Partial Fulfillment to the Requirements
for the Award of the Degree of Doctor of Philosophy
in Computer Engineering

December 2022

ABSTRACT

NEURAL SEMANTIC PARSING, ANNOTATION AND EVALUATION FOR TURKISH

NECVA BÖLÜCÜ
Doctor of Philosophy , Computer Engineering
Supervisor: Assoc. Prof. Dr. HARUN ARTUNER
2nd Supervisor: Assoc. Prof. Dr. BURCU CAN BUĞLALILAR
December 2022, 220 pages

Semantic representation is a way of expressing the meaning of a text that can be processed by a machine to serve a particular natural language processing (NLP) task. Universal Conceptual Cognitive Annotation (UCCA) is one such semantic representation form that is both cognitively and linguistically inspired. UCCA represents the meaning of a text with a directed acyclic graph (DAG) whose nodes can be either terminal or non-terminal nodes, where terminal nodes correspond to tokens and multi-tokens in the text, non-terminal nodes comprise several tokens that are jointly viewed as a single entity according to some semantic or cognitive consideration, and edges indicate the role of a child in a relation. In this thesis, there are three research paths within UCCA representation especially for Turkish language: semantic parsing, data annotation, and evaluation of UCCA representation as extrinsic evaluation in other NLP problems.

In the first part of the thesis, we present supervised deep learning-based parsing models, which are transition and graph-based approaches, to better analyze the approaches for

UCCA representation. We also present an unsupervised deep learning model that leverages pre-trained language models (PLM) as an external knowledge source.

In the second part of the thesis, we present a Turkish UCCA-annotated dataset, that is built using the proposed graph-based semantic parser in a semi-automatic pipeline.

Finally, we investigate using UCCA for other NLP tasks including Semantic Textual Similarity (STS), text classification, and question answering (QA) as extrinsic evaluation of UCCA representation. It is therefore reasonable to ask whether we can improve the performance of NLP tasks by using semantic information in the form of UCCA representation. In conclusion, the results show that semantic information in the form of UCCA representation improves performance in NLP tasks, especially in tasks that require more semantic information, such as QA.

Keywords: UCCA, semantic representation, Turkish, parsing, annotation, evaluation

ÖZET

TÜRKÇE İÇİN NÖRAL SEMANTİK AYRIŞTIRMA, ETİKETLEME VE DEĞERLENDİRME

NECVA BÖLÜCÜ
Doktora, Bilgisayar Mühendisliği
Danışman: Doç. Dr. Harun ARTUNER
Eş Danışman: Doç. Dr. Burcu Can BUĞLALILAR
Aralık 2022, 220 sayfa

Semantik gösterim, bir doğal dil işleme uygulamasında kullanılmak üzere verilen bir metnin makine tarafından işlenebilecek şekilde anlamını ifade etmek için bir yöntem sağlar. Hem bilişsel hem de dilsel temellere dayanarak tasarlanan Universal Conceptual Cognitive Annotation (UCCA) bu semantik gösterimlerden sadece birisidir. UCCA'da bir metnin anlamı döngüsüz çizge ile ifade edilmektedir ve çizgede düğümler terminal veya terminal olmayan düğümlere ve kenarlar çocuk düğümün çizgedeki rolüne karşılık gelmektedir. Terminal düğüm metindeki sözcük yada birden çok sözcüğe karşılık gelirken, terminal olmayan düğüm semantik yada bilişsel değerlendirmelere göre tek bir birim olarak görülen birkaç sözcüğe karşılık gelmektedir. Bu tezin, özellikle Türkçe UCCA gösterimi altında 3 odak noktası bulunmaktadır: semantik ayrıştırma, veri etiketleme ve UCCA gösteriminin diğer NLP problemlerinde harici olarak kullanıp değerlendirilmesi.

Tezin ilk bölümünde, UCCA gösterimi için yaklaşımları daha iyi analiz edebilmek için, denetimli derin öğrenme tabanlı ayrıştırma modellerinden geçiş ve çizge tabanlı yaklaşımlar açıklanmaktadır. Ayrıca, harici kaynak olarak önceden eğitilmiş dil modellerinden yararlanan denetimsiz bir derin öğrenme modeli de tanıtılmaktadır.

Tezin ikinci bölümünde, yarı-özdevinimli olarak etiketlenmiş Türkçe UCCA veri kümesi tanıtılmaktadır. Veri etiketleme esnasında çizge modelinden faydalanılmıştır.

Son olarak, tez UCCA gösteriminin harici olarak değerlendirilmesi amacıyla semantik metinsel benzerlik, metin sınıflandırma ve soru cevaplama dahil olmak üzere diğer doğal dil işleme problemlerinde kullanılmış ve sonuçlar tartışılmıştır. Burada temel soru, UCCA gösteriminin semantik bilgi için kullanılmasının doğal dil işleme uygulamalarında performası iyileştirip iyileştiremeyeceği yönündedir. Sonuç olarak, UCCA semantik gösteriminin doğal dil işleme uygulamalarının, özellikle daha fazla semantik bilgi gerektiren uygulamalarının performansını iyileştirdiği gözlenmiştir.

Keywords: UCCA, semantik gösterim, Türkçe, ayrıştırma, etiketleme, değerlendirme

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my supervisor Dr. Burcu Can for her invaluable guidance, wisdom, patience, and unlimited support since my master years. I would also like to thank my supervisor Assoc. Prof. Dr. Harun Artuner for his guidance and support. Furthermore, I would like to thank the members of my thesis committee Prof. Dr. Hüseyin Cem Bozşahin, Assoc. Prof. Dr. Sinan Kalkan for their invaluable feedback and generously provided knowledge and expertise.

I would also like to extend my thanks to research assistants Merve Özdeş, Bahar Gezici, Nebi Yılmaz, Asst. Prof. Dr. Pelin Canbay, Burçak Asal, Zeynep Bala, and Burcu Yalçıner from the Department of Computer Engineering at Hacettepe University, and Pınar Uskaner Hepsağ and Gizem Karaca from the Department of Computer Engineering at Alparslan Türkeş Science and Technology University for their support and understanding. I am deeply indebted to Zehra Karhan, Dr. Yunus Can Bilge, and Dr. Selma Dilek for their support and motivation.

Finally, I would like to thank my parents for always supporting me in my life and my family for motivating me with their great support and patience for my studies. Without their understanding and encouragement, I could not keep my spirits and motivation high.

CONTENTS

TABLES

Page

FIGURES

ABBREVIATIONS

AM	:	**A**pply **M**odify
AMR	:	**A**bstract **M**eaning **R**epresentation
BLT	:	**B**asic **L**inguistic **T**heory
BiLSTM	:	**Bi**directional **L**ong **S**hort **T**erm **M**emory
CA	:	**C**onstituent **A**ttention
CNN	:	**C**onvolutional **N**eural **N**etwork
CYK	:	**C**ocke **Y**ounger **K**asami
DAG	:	**D**irected **A**cyclic **G**raph
DAM	:	**D**ecompositional **A**ttention **M**odel
DL	:	**D**eep **L**earning
EDS	:	**E**lementary **D**ependency **S**tructures
EM	:	**E**xact **M**atch
GAT	:	**G**raph **A**ttention **N**etwork
GCN	:	**G**raph **C**onvolutional **N**etwork
GNN	:	**G**raph **N**eural **N**etwork
HRG	:	**H**yperedge **R**eplacement **G**rammar
IMST	:	**I**TU **M**ETU **S**abancı **T**reebank
IR	:	**I**nformation **R**etrieval
kNN	:	**k** **N**earest **N**eighbor
LR	:	**L**ogistic **R**egression
LSTM	:	**L**ong **S**hort **T**erm **M**emory
ML	:	**M**achine-**L**earning
MLEC	:	**M**ulti-**L**abel **E**motion **C**lassification
MLP	:	**M**ulti-**L**ayer **P**erceptron
MRP	:	**M**eaning-**R**epresentation **P**arsing
MST	:	**M**etu **S**abancı **T**reebank

NLP	:	**N**atural **L**anguage **P**rocessing
NLU	:	**N**atural **L**anguage **U**nderstanding
NMT	:	**N**eural **M**achine **T**ranslation
OOV	:	**O**ut-**O**f-**V**ocabularu
PASCAL	:	**PA**rent-**SCAL**ed Self-Attention
PNG	:	**P**lain **G**raph **N**otation
PLM	:	**P**tretrained **L**anguage **M**odel
PMB	:	**P**arallel **M**eaning **B**ank
RC	:	**R**eading **C**omprehension
RE	:	**R**elation **E**xtraction
RNN	:	**R**ecurrent **N**eural **N**etwork
RvNN	:	**R**ecursive **N**eural **N**etwork
SAWR	:	**S**yntax-**A**ware **W**ord **R**epresentation
SDP	:	bilexical **S**emantic **D**e**p**enencies
SGSA	:	**S**yntax-**G**raph **S**elf **A**ttention
SLA	:	**S**yntax-aware **L**ocal **A**ttention
SRL	:	**S**emantic **R**ole **L**abeling
STS	:	**S**emantic **T**extual **S**imilarity
SVM	:	**S**upport **V**ector **M**achine
UCCA	:	**U**niversal **C**onceptual **C**ognitive **A**nnotation
UDS	:	**U**niversal **D**ecompositional **S**emantics
QA	:	**Q**uestion **A**nswering

1. INTRODUCTION

Semantics is concerned with everything that is related to meaning. In linguistics, a distinction is made between semantic representation and semantics, the former being more concerned with relations between the text components; i.e., words, statements, etc. [14]. Semantic representation is a way to transform the meaning of a natural language utterance so that it can be understood by machines in much the same way that humans understand natural language. With the increasing attention to semantic representation, a number of graph-based semantic representation frameworks have been proposed, such as Abstract Meaning Representation (AMR) [15], Elementary Dependency Structures (EDS) [16] and Universal Conceptual Cognitive Annotation (UCCA) [17], Universal Decompositional Semantics (UDS) [18]. These semantic representations facilitate meaning to be pervasively used in a structured form in Natural Language Processing (NLP) and Natural Language Understanding (NLU) applications, such as text summarization [19–21], paraphrase detection [22], neural machine translation (NMT) [23–25], question answering (QA) [26–28], and text simplification [29].

UCCA is such a graph-based semantic representation, that is based on cognitive theories [30] and consists of multiple layers that allow abstracting from syntactic forms by representing only semantic distinctions and treating syntax as a hidden layer in learning the mapping between form and meaning. UCCA is represented as a directed acyclic graph (DAG) whose edges are labeled with the UCCA categories (semantic relations) and whose nodes refer to units that are either terminal or non-terminal nodes, where terminal nodes are tokens and non-terminal nodes compromising several tokens that are jointly considered as a single entity according to semantic or cognitive consideration. Moreover, UCCA is not based on English grammar and, unlike other semantic annotations such as AMR, does not require syntactic preprocessing for learning. AMR is also a well-studied semantic representation. However, it requires PropBank [31] predicates as edges in the annotation. All of these properties contribute to the choice of the UCCA representation as the focus of this thesis.

With annotated data, it becomes conceivable to train deep learning (DL) models to map the data in their UCCA representation. In the first part of this thesis, we develop algorithms for learning to map natural language to UCCA using released UCCA datasets [1]. In order to understand the unique properties of the UCCA formalism, that brings new challenges to the parsing community, namely reentrancy, non-terminal nodes, implicit units, and discontinuous units, we develop supervised and unsupervised algorithms to understand the limitations of these models for the problem by using different approaches and comparing the unsupervised model with 3 different parsing problems, namely constituency parsing, dependency parsing, and semantic parsing. This thesis contributes in the form of novel approaches to parsing and deep analyses of the models.

UCCA annotated datasets have been released so far for English, French, German, Russian, and Hebrew. However, semantically annotated datasets in Turkish, a low-resource language in this field, are very scarce. In the literature, only the AMR dataset exists as a semantic dataset for Turkish [32]. The parsing models are applicable in a pipeline for the annotation process of the datasets. In the second part of this thesis, we used one of the semantic parsing models we developed to annotate a UCCA dataset for Turkish in a semi-automatic annotation process. Since Turkish is an agglutinative language and has complex grammar rules compared to other languages such as English and German, a detailed analysis of the rules is required for annotation. Using the output of the parser model on the Turkish dataset, we refine the discrepancies between the output of the parser model and the Turkish guideline. The other contribution of the thesis is the released Turkish UCCA dataset. We believe that this dataset will be a crucial resource for advancing the state of the art in semantic parsing in Turkish and Turkish UCCA parsing in particular.

It is possible that UCCA significantly advances the state-of-the-art in NLP tasks because UCCA captures relations between nodes and represents abstract semantic meaning. It can be used as an intermediate representation in a variety of tasks that require semantic meaning, such as semantic textual similarity (STS), QA, and sentiment analysis. In the third and final part of the thesis, we develop semantic-aware models for various NLP problems and compare the models using UCCA graphs with other semantic and syntactic representations for NLP problems. The contribution of the final part is semantic-aware models for NLP problems with a deep analysis of the representations in the models.

1.1. Scope of the Thesis

UCCA has gained popularity because it is portable across domains and languages and covers sentences or paragraphs with more than one sentence. The framework is also beneficial in many NLP tasks, such as summarisation [33, 34] and QA [26, 35]. In addition, there has been increasing popularity in developing semantic-aware models, that use semantic representations to capture the abstract meaning of a sentence using long-distance semantic information. Unfortunately, despite these benefits of the UCCA framework, there is still a gap in semantic representations in Turkish.

In this thesis, the UCCA framework, which allows easy annotation without requiring substantial linguistic knowledge, is explored in terms of parsing models, the annotation process, and evaluation in other tasks, mainly for Turkish. The goal of this research is to propose novel DL based semantic parsing models for the UCCA framework, focusing on supervised and unsupervised methods. Moreover, the proposed semantic parser is used for a semi-automatic pipeline for Turkish UCCA annotation, which will be a valuable resource for NLP applications in Turkish. Finally, the UCCA framework is evaluated by using it as external information in the DL models for various NLP problems.

To the best of our knowledge, there is no previous research that has introduced an unsupervised parser model for UCCA parsing by comparing the model for different levels of parsing from syntax to semantics. Moreover, there has been no UCCA dataset for Turkish before. We claim that the dataset is of paramount importance for improving performance and implementing high-quality systems. The studies dealing with NLP tasks have neglected to thoroughly investigate the semantic representation to improve the performance of the models. In addition to the datasets, this is the first attempt to develop semantic-aware DL models using the UCCA framework for different NLP tasks and to compare semantic and syntactic representations for these tasks.

1.2. Main Contributions of the Thesis

In this thesis, we address the deficiencies in the literature for the UCCA framework. We propose different parsing methods for the UCCA framework by applying different approaches and supervised settings. Since there is no annotated Turkish UCCA dataset, we apply one of the proposed methods to ease the annotation process. To address the semantic information in NLP problems, we propose novel models that integrate the UCCA framework into the input of the models. The main contributions of this thesis are as follows:

- We propose a graph-based and a transition-based approach to compare the approaches on the UCCA framework. The graph-based approach is a self-attentive parser that addresses the parsing problem in the form of chart-based constituency parsing. The transition-based approach is an incremental parser where the transition set is updated to cover the properties of the UCCA framework.
- By conducting an experimental analysis, we demonstrate the effectiveness of the proposed graph-based approach compared to the incremental parser. We also conducted the experiments in zero-shot and few-shot settings to understand the effect of the size of the dataset, since the size of the dataset is smaller for some languages than for others. We also applied single-lingual and cross-lingual experiments for in-depth analysis in terms of the dataset.
- We also propose an unsupervised parser called the chart-based zero-shot parsing model, which uses a pre-trained language model (PLM) and is the first attempt at fully unsupervised UCCA parsing. We adopted the method for the constituency and dependency parsing in addition to UCCA parsing to evaluate various parsing levels using the same model and to understand the behavior of the model for different levels of parsing from syntax to semantics.
- We propose a pipeline process for data annotation, where we use a parser method to build the first UCCA dataset in Turkish. We define UCCA annotation rules, which are not covered in the English UCCA guideline, in line with the Turkish syntax

- We propose a DL model for Semantic Textual Similarity (STS) using the UCCA framework. Moreover, we used the results of the model to generate semantically informed sentence embeddings that can be used in the downstream tasks.
- We use the semantic representation as input for text classification in different domains and with different labels. We apply the AMR semantic framework and dependency parsing to evaluate the limitations of the UCCA-based model.
- We also propose a semantic informed model for Question Answering (QA) by incorporating deep semantic information and capturing long distance information.

1.3. Organization

The organization of the thesis is as follows:

- Chapter 1. gives a brief introduction to the problem, presents our motivation for the next chapters, the main contributions, and the scope of the thesis.
- Chapter 2. provides background on relevant topics, including semantic representations, UCCA framework, and semantic parsing approaches.
- Chapter 3. provides detailed background on Turkish grammar.
- Chapter 4. introduces the proposed semantic parser models. First, we introduce the self-attentive parser, which is a graph-based approach. Then, we introduce the incremental parser, which is a transition-based approach. Finally, we present an unsupervised chart-based zero-shot parser, which is a fully unsupervised model. We also present the experiments we performed on the released UCCA datasets for the models and the Turkish UCCA dataset annotated in this thesis and analyse the results.
- Chapter 5. presents the complete annotation process and describes how an external semantic parser has been used in the annotation of the UCCA framework. First, we present the extracted rules for the Turkish grammar. Finally, the quality of the created dataset is demonstrated by annotation agreement and annotation statistics.

- Chapter 6. demonstrates the proposed models for the NLP problems STS, text classification, and QA used in the extrinsic evaluation of UCCA, mainly for Turkish. We also present the experiments we conducted for the proposed models, with detailed experimental analyses to demonstrate the effectiveness of the proposed models.
- Section 7. concludes this study with a summary of the thesis and possible future work.

2. BACKGROUND

2.1. Semantic Representations

Semantics is concerned with meaning defined by relations between words in a sentence. Revealing semantic relations between words or a group of words in a sentence helps to better understand natural languages, and this will eventually help in the development of linguistically motivated semantic models in NLP applications. While semantics is the study of the relations between the building blocks of a sentence (i.e., words or a group of words) and their implicit meaning, semantic representation reflects the meaning of the text in a rather structured form (e.g., graph-based or tree-based representation) [14].

Graphs have been receiving increasing attention in NLP in recent years due to their ability to express and generate adequate target structures, especially for sentence-level syntactic analysis and semantic representation of a text. The increasing popularity of graph-based semantic representations has led to the proposal of various semantic representation frameworks such as AMR [15], UCCA [17], bilexical Semantic Dependencies (SDP) [36], UDS [18], and Parallel Meaning Bank (PMB) [37].

Kuhlmann and Oepen [38] define 3 types: Flavor(0), Flavor(1), and Flavor(2) for a formal graph to distinguish graph-based semantic frameworks by specifying the form of anchoring in graphs. If we sort the Flavors from the strongest to the weakest form of anchoring, Flavor(0) indicates the strongest form of anchoring, where each node is directly linked to a specific token. Flavor (1) is a more general form of an anchored semantic graph, where there is a more relaxed anchoring that allows linkage between nodes and arbitrary parts of the sentence, which may be a sub-token or multi-token sequence. This provides flexibility in the representation of meaning. EDS [16] and UCCA [17] are categorized in Flavor(1). UCCA defines semantic graphs with a multi-layer structure, where the foundational layer of representation focuses on the integration of all surface tokens into argument structure phenomena (e.g., verbal, nominal, adjectival). The terminal nodes of a graph are anchored to possibly discontinuous sequences of surface substrings, whereas the interior nodes of the graph are not. Finally, Flavor(2) indicates unanchored graphs where there is no correspondence between nodes and tokens. AMR [15] is categorized in Flavor(2), a sentence-level semantic representation framework that uses unanchored graphs where the nodes represent concepts (predicate frames, special keywords, etc.) and the edges represent semantic relations between them to avoid explicit mapping of graph elements to surface utterances.

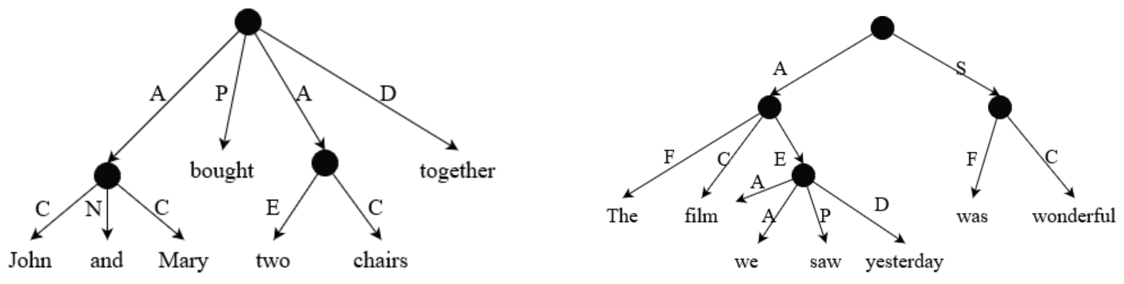
We give the details of the UCCA representation, which is the focus semantic representation of this thesis.

2.2. Universal Conceptual Cognitive Annotation (UCCA)

UCCA is a semantic annotation scheme introduced by Abend and Rappoport [17] in 2013. It is a cross-linguistic annotation scheme for encoding semantic annotations with a multi-layered framework in which each layer corresponds to a "module" of semantic distinctions. The foundational layer of UCCA focuses on all grammatically relevant information, including predicate-argument relations to predicates of all grammatical categories.

We focus on the foundational layer of UCCA described in Abend and Rappoport [17, 39] which focuses on grammatically relevant information. The aim of the representation is to represent the main semantic phenomena in a text by abstracting from syntactic forms, while maintaining low learning cost and rapid annotation by non-experts [14]. UCCA is a cross-linguistic representation that has proven effective in several languages and is a popular target framework that has been studied in multiple parsing tasks [1, 40]. The effectiveness and efficiency in annotating and refining UCCA has been demonstrated in several studies [41, 42].

UCCA is represented by a directed acyclic graph (DAG) whose leaves correspond to tokens and multi-tokens in the text. The nodes of the graph are known as units, which are either terminals or non-terminals. Multiple tokens correspond to a single entity based on a particular semantic or cognitive consideration. The edges of a graph refer to the categories



(a) The annotation of the sentence *"John and Mary bought two chairs together"*

(b) The annotation of the sentence *"The film we saw yesterday was wonderful"*

Figure 2.1 Examples of UCCA annotation graphs

of its children. In the UCCA representation, there are 4 main categories: (i) Scene Elements: Process (P), State (s), Participant (A), Adverbial (D), (ii) Non-Scene unit Elements: Center (C), Elaborator (E), Connector (N), Relation (R), (iii) Inter-Scene relations: Parallel Scene (H), Linker (L), Ground (G), and (iv) Other: Function (F).

Two sentences annotated based on the UCCA framework are given in Figure 2.1. In the sentence given in Figure 2.1(a), there is a Scene with a relation called **Process**, which corresponds to *"bought"*. *"John and Mary"* and *"two chairs"* are the **Participant**s of the Scene, and *"together"* is the **Adverbial** in this Scene. The **Participant** *"John and Mary"* consists of entities of the same type that is called **Center** for both *"John"* and *"Mary"*, connected by *"and"* which is a **Connector**. The **Participant** *"two chairs"* is composed of a **Center** that is *"chairs"* and an **Elaborator** that is *"two"*, which describes the **Center**. In the second sentence given in Figure 2.1(b), there is a Scene that contains a relation called **State** and a **Participant**. The **State** consists of a **Function** and a **Center** and the **Participant** consists of an **Elaborator**, a **Function**, and a **Center**. The **Elaborator** *"film we saw yesterday"* is an **E-Scene**, because *"saw"* evokes another **Scene**. While *'film"* is the **Center** of the **Participant**, it also serves as the **Participant** of the **E-Scene**, resulting in *'film"* has two parents.

2.2.1. UCCA Categories

The foundational layer views the text as a collection of `Scenes`. A Scene describes a movement or an action, or a temporally persistent state. It usually contains information regarding when the Scene happens, the location, and the ground that explains how it happens[1]:

Examples:

- Alex played at school (1 Scene)
- Alex went home and started cooking (2 Scene)

2.2.1.1. Scene Elements A Scene has only one main relation, which determines the type of the Scene (either dynamic or static). The Relation becomes a `Process` (P) if there is an action or a movement. However, if a Scene evolves in time, it becomes a `State` (S), which means that the Scene describes a temporally persistent state.

Examples:

- Alex is $\langle$weighted$\rangle_S$
- Alex $\langle$kicked$\rangle_P$ the ball
- It is $\langle$in$\rangle_S$ the school

A Scene contains one or more `Participants` (A). They can be concrete or abstract. Embedded Scenes are also considered as the Participant of the main Scene.

Examples:

- $\langle$Alex$\rangle_A$ is weighted

[1]Please note that no punctuation is given in the example sentences for simplicity.

- $\langle$It$\rangle_A$ is in $\langle$the school$\rangle_A$

The secondary relations of the Scenes are marked as `Adverbials` (D). They describe the main relation (P/S) in the Scene. It can refer to the time, frequency, duration of the process or state, as well as modality in verbs (e.g., can, will, etc.), the manner relations, and relations that specify a sub-event (begin, end, finish etc.).

Examples:

- Alex $\langle$began$\rangle_D$ playing the guitar
- Alex $\langle$may$\rangle_D$ sleep earlier today
- Alex $\langle$may$\rangle_D$ come at around eight$_D$
- He saw the doctor $\langle$regularly$\rangle_D$

2.2.1.2. Non-Scene Unit Elements There are also non-Scene relations in the UCCA framework that do not evoke a Scene. Each non-unit contains one or more `Center` (C), which is required for the conceptualization of the non-Scene unit. It is the main element of the non-Scene unit, and other relations may elaborate or be associated with this element. Class descriptors that determine the semantic type of the parent unit are considered as an `Elaborator` (E) of the main element. Quantifiers describing the quantity of the magnitude of an entity or expression are also identified as an `Elaborator` (E):

Examples:

- $\langle$chocolate$\rangle_E$ $\langle$cookies$\rangle_C$
- I bought $\langle\langle\langle$three$\rangle_E$ $\langle\langle$kilos$\rangle_C\rangle_E$ $\langle\langle$of$\rangle_R$ $\langle$apples$\rangle_C\rangle_E\rangle_A$
- $\langle$Queen$\rangle_C$ $\langle\langle$of$\rangle_R$ $\langle$England$\rangle_C\rangle_E$
- $\langle$big$\rangle_E$ brown$_E$ dog$_C$

- $\langle$three$\rangle_E$ $\langle$apples$\rangle_C$

`Connectors` (N) are the relations between two or more entities with similar features or types.

Example:

- I will have $\langle\langle$coffee$\rangle_C$ $\langle$and$\rangle_N$ $\langle$cookies$\rangle_C\rangle_A$

The other type of relation between two or more units is called `Relator` (R), which does not evoke a Scene. In two different scenarios, the relation is marked as a Relator:

1. A single entity is related to the other relation in the same context as a Relator. In this case, the Relator should be positioned as a sibling of the C (or the Scene). It is placed inside the unit they pertain to:

   Examples:

   - Alex said [$\langle$that$\rangle_R$ he'll not come to the party]
   - Alex met [$\langle$with$\rangle_R$ $\langle$Mary and Jane$\rangle_C$]

2. Two units attached with different aspects of the same entity are related through a Relator.

   Example:

   - $\langle\langle$bottom $\langle$of$\rangle_R\rangle_E$ $\langle\langle$the$\rangle_E$ $\langle$sea$\rangle_C\rangle_C$

Linkage is the term used for Inter-Scene relations in the UCCA. There are four types of Linkages in the UCCA adopted from the Basic Linguistic Theory [43]:

1. Elaborator Scenes: `E-Scene` adds information to a previously established unit to answer the questions `which X` or `what kind of X`.

Examples:

- [The cat $\langle$that played with my child $\langle$(cat)$\rangle_A\rangle_E$] is brown
- My fried $\langle$whom I gave [the present] [$\langle$to$\rangle_R$ (friend)]$\rangle_E$
- Alex played [an American $\langle$taken to the Adriatic (American)$\rangle_E$]

2. Center Scenes: `C-Scene` is a `Center` unit of a larger Scene that is also a Scene that is internally annotated.

   Examples:

   - the $\langle$taxi driver$\rangle_C$ who swims
   - the tall $\langle$English teacher$\rangle_C$ arrived

3. Participant Scenes: `A-Scene` is a participant in a Scene and has a removable role, as it does not add information to the participant in the main Scene and answers the question `what` in a Scene.

   Examples:

   - $\langle$Drinking cold water$\rangle_A$ is ill-advised (what is ill-advised)
   - Alex said $\langle$he's thursty$\rangle_A$ (what did Alex say)
   - $\langle\langle$Alex 's$\rangle_A$ wonderful slam dunk$\rangle_A$ saved the game (what saved the game)

4. Parallel Scenes: If a Scene is not a `Participant`, `Center`, or `Elaborator` in a Scene and is connected to other Scenes by a `Linker`, which is a relational word between Scenes, the Scenes are called `Parallel Scenes` (H):

   Examples:

   - $\langle$Alex played basketball$\rangle_H$ $\langle$while$\rangle_L$ $\langle$preparing for exams $\rangle_H$
   - $\langle$After$\rangle_L$ $\langle$graduation$\rangle_H$, $\langle$Alex moved to NYC$\rangle_H$

- $\langle$When$\rangle_L$ $\langle$I arrived at the train station$\rangle_H$ $\langle$I realised that I had forgotten my cell phone$\rangle_H$
- $\langle$If$\rangle_L$ $\langle$they called us$\rangle_H$ $\langle$we will certainly go to the party$\rangle_H$
- $\langle$I completed my assignment$\rangle_H$ $\langle$asked my questions to my professor$\rangle_H$ $\langle$and$\rangle_L$ $\langle$started working on a new project$\rangle_H$
- $\langle$You knew the answer of this question only$\rangle_H$ $\langle$because$\rangle_L$ $\langle$I explained it to you$\rangle_H$

`Ground` (G) is a unit used to relate some unit to its speech event; either the speaker, the hearer, or the general context in which the text was uttered/written/conceived. It is similar to `Linker` except that it relates the Scene to the speech act of the text (the speaker, the hearer, or their opinions).

Examples:

- $\langle$Interestingly$\rangle_G$ our team won the game
- $\langle$In my opinion$\rangle_G$, Alex won't come to the party
- $\langle$I was shocked$\rangle_H$ $\langle$when$\rangle_L$ our team lost the game

2.2.2. Other

There are also `Function` (F) units in which the terminals do not refer to a participant or a relation. They function only as part of the construction in which they are situated in:

Examples:

- I want $\langle$to$\rangle_F$ run [a marathon]
- It $\langle$is$\rangle_F$ likely that$_F$ Alex will come
- $\langle$Let$\rangle_F$ me introduce Alex

2.2.3. Remote and Implicit Units

While some relations are clearly described in the text, there are instances where a sub-unit in a given unit is not explicitly mentioned:

1. If the missing entity is referenced from another position in the text, we add a reference to the missing unit, which is labeled as a `REMOTE` unit (the minimal unit is used as REMOTE).

   Examples:

   - [Alex went home] and [started cooking $\langle$(Alex)$\rangle_{A-REMOTE}$]
   - [The cat [that played with my child $\langle$(cat)$\rangle_{A-REMOTE}$]] is brown

2. When the missing entity does not appear in any place in the text, we add an `IMPLICIT` entity that stands for the missing sub-unit.

   Examples:

   - We just opened $\langle$(the thing)$\rangle_{A-IMPLICIT}$
   - [Alex is tall], [Mary is $\langle$(tall)$\rangle_{C-IMPLICIT}$ n't]

In general, UCCA is built based on the following principles:

- **Graph Representation** UCCA is represented by a directed, edge-labeled acyclic graph, which enables reentrancy. UCCA is defined as a multi-layer structure that allows an unbounded extension. It also enables paragraph-level annotation.

- **Multiple Parents** In the UCCA representation, a unit may participate in more than one relation, so it has multiple parents in the graph. This relation is represented by remote edges. UCCA distinguishes between primary edges, which appear explicitly in a relation, and remote edges, which allow a Scene to indicate its arguments by linking from another Scene. Primary edges form a tree, while Remote edges allow reentrancy and form a DAG.

- **Non-terminal Nodes** Unlike other representations, the UCCA uses non-terminal nodes to represent units compromising more than one word.
- **Discontinuity of Semantic Units** In the UCCA representation, some units in the text may be discontinuous. Discontinuity that is a property of UCCA, is pervasive e.g., with multi-word expressions. For example, in “Alex gave everything up”, the phrasal verb *“gave ... up”* forms a single discontinuous semantic unit.
- **Implicit Units** UCCA also distinguishes an entity that is important to the interpretation of a Scene but does not exist explicitly in the text. UCCA introduces Implicit Units to represent such kind of entity.

2.3. UCCA Parsing

UCCA parsing is the task of converting natural language into UCCA graphs. Depending on their formal properties, various parsing methods have been used for various frameworks. The main parsing approaches can be categorized as *transition-based* and *graph-based.*

2.3.1. Transition-based Parsing

Transition-based parser [44, 45], also called shift-reduce parser, is a general approach to building a tree or graph from a natural language. The parse tree or graph is built by applying a *transition* at each step to the parser state defined by a buffer *B* of tokens and nodes to be processed, a stack *S* of processed nodes and a graph $G = (V, E, l)$ of constructed nodes and labeled edges, where V is the set of *nodes*, E is the set of *edges*, and $l : E \to L$ is the *label* function, where L is the set of possible labels. At each step, a classifier is used to select the next transition based on the features that encode the current state of the parser. Starting from an initial state, the parser applies actions until a terminal state, containing the final parse, is reached.

The first transition-based parser proposed for generating UCCA representations is the TUPA parser [46]. This is an extension of the standard transition-based parser for handling UCCA properties, reentrancy, discontinuity, and remote nodes. The operations of the transition-based parser are extended by adding additional operations to the standard transition-based parser operations `SHIFT`, moving a node from the buffer to the stack *S* and `REDUCE` the possible appending of nodes and removing nodes from the stack *S* [47]. The additional operations are $\texttt{Node}_X$, $\texttt{Left-Edge}_X$, $\texttt{Right-Edge}_X$, $\texttt{Left-Remote}_X$, $\texttt{Right-Remote}_X$, `Swap`, and `Finish` [46].

In the training step, an oracle uses the gold-standard annotation to create training instances for the classifier and the parser uses the instances to reconstitute the representation graph in a deterministic way.

2.3.2. Graph-based Parsing

Graph-based parser [48] is a general approach based on building a tree or graph by performing the entire parsing process as graph operations. Graph-based parsers parameterize a model over smaller substructures to search the space of valid graphs and generate the most likely one. The simplest parameterization is the arc-factored model, which defines scores for spans $s(i, j, l)$, where i and j denote the starting and ending positions of the span with label l from the label set L, and also defines the score of a graph as the sum of the scores of all spans contained in it.

A UCCA graph for an input sentence $w_0, \cdots, w_n$ is a labeled directed graph $G = (V, E)$ consisting of a set of nodes (terminal or non-terminal) and a set of labeled edges $E \subseteq V \times V \times L$, where L is the set of possible labels. The graph-based parsing problem is equivalent to finding the directed spanning tree with the highest score in the complete graph over the input sentence, as defined below:

$$\hat{T} = \underset{G=(V,E)}{\arg\max} \sum_{(i,j,l)\in E} s(i, j, l) \tag{1}$$

2.3.3. Composition-based Parsing

Composition-based approach [49, 50] is an approach that builds a graph with a set of lexical and syntactic-semantic rules governed by a well-defined grammar formalism such as Hyperedge Replacement Grammar (HRG) [49] and Apply-Modify (AM) Algebra [50].

Composition-based parsers model derivations yielding semantic graphs by defining a score function $SCORE_D$ for $D = r_1, r_2, \cdots, r_m$ a set of rules. The parsing problem is equivalent to finding rules with the highest score to find the graph over the input sentence as defined below:

$$G' = \underset{G* \in GEN(x)}{\arg\max} \sum_{D \in DERIV(G*)} SCORE_D(D) \tag{2}$$

In the scope of this thesis, we have presented three semantic parser models in Chapter 4., which are transition-based and graph-based approaches. Self-attentive and chart-based zero-shot parsing models, which are graph-based approaches, solve the parsing problem as constituency parsing using CYK to build semantic graphs. Incremental parser, which is a transition-based approach, applies a new transition set to the semantic parsing problem.

2.4. Other Semantic Representations

Due to the increasing popularity of graph-based semantic representations, several frameworks for semantic representations have been proposed in the literature. Abstract Meaning Representation (AMR), the bilexical Semantic Dependencies (SDP), and Parallel Meaning Bank (PMB) are among these representations.

2.4.1. Abstract Meaning Representation (AMR)

AMR was introduced by Banarescu et al. [15] and represents a semantic representation language that captures the meaning of a sentence with the question *"who is doing with whom"* in the sentence. AMR represents semantic representation using a graph, which is a

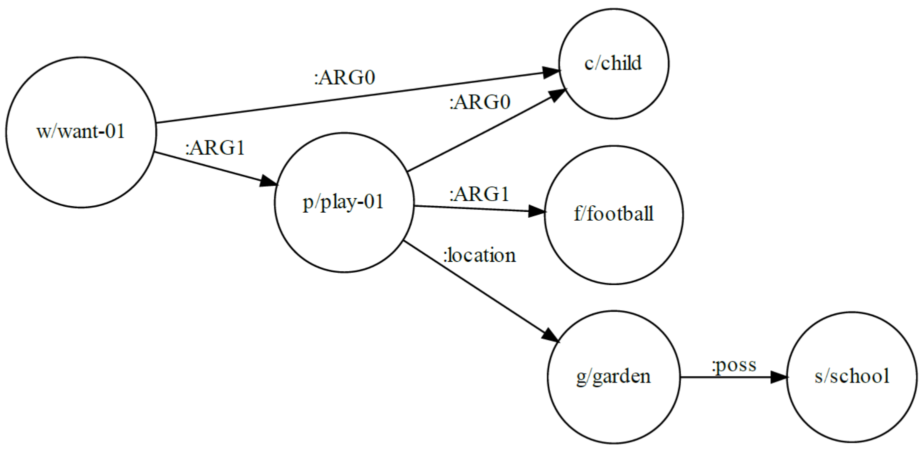


Figure 2.2 Example of AMR annotation graph

rooted directed acyclic graph (DAG) in which nodes represent core concepts in the sentence, which can be either word (adjectives, stemmed nouns, and adverbs) or frame extracted from PropBank [51], and labeled edges between nodes indicate a semantic relationship between nodes. Edges can have either core roles (PropBank predicates) indicating semantic roles between AMR concepts, or non-core roles such as location, time, purpose, etc., which form the rest of AMR relations. AMR drops inflectional morphology that defines tenses and omits articles and punctuation that do not give logical meaning to graphs (including a logical rather than syntactic representation). The aim of AMR representation is to give the same representation to sentences that have the same meaning (e.g., the sentences "Children wanted to go to school" and "Children's wish is to go to school" may have the same AMR representation). An example of the AMR representation of the sentence *"Children want to play football at the garden of the school."* is shown in Figure 2.2. For more details on AMR, see the AMR guidelines [15].

2.4.2. Bilexical Semantic Dependencies (SDP)

Bilexical Semantic Dependencies (SDP) was introduced by Oepen et al. [36] and represents a semantic representation that is an extension of the previous version used as Semantic Dependency Parsing representations in SemEval 2014 [52] and 2015 [53]. SDP representations are characterized by a graph $G = (V, E)$, where V is a set of nodes with labels using word form, optional lemma, part of speech, a Boolean flag indicating whether the node represents a top predicate, and optional frame tag and $E \in V \times V$ is a set of edges labeled with semantic relations that hold between their source and target nodes.

2.4.3. Parallel Meaning Bank (PMB)

Parallel Meaning Bank (PMB) was introduced by [37], a cross-lingual representation, that provides fine-grained meaning representations for words, sentences, and texts. The PMB representation has 8 annotation layers ranging from syntactic to semantic (segmentation, symbolization, word sense disambiguation, syntactic parsing, semantic role labeling, semantic tagging, coreference resolution, and semantic parsing.

3. TURKISH GRAMMAR

In this chapter, we briefly describe Turkish grammar along with the morphological and syntactic features of the language that were consulted during the annotation process.

3.1. Morphology

Turkish is an agglutinative language where suffixes are attached to word roots or stems to form words, known as suffixation. Therefore, Turkish words commonly contain more than one syllable [54]. An example of suffixation for the root *kitap (book)* is given below:

$defter$	$\rightarrow$	*notebook*
$defter -$ **ler**	$\rightarrow$	*notebook**s***
$defterler -$ **imiz**	$\rightarrow$	***our** notebooks*
$defterlerimiz -$ **de**	$\rightarrow$	***on** our notebooks*
$defterlerimizde -$ **ki**	$\rightarrow$	***the one** on our notebooks*
$defterlerimizdeki -$ **ler**	$\rightarrow$	*the one**s** on our notebooks*

3.1.1. Morphemes

Morphemes are defined as the smallest meaning-bearing units in a language. There are two types of morphemes, depending on whether they are attached to a word or stand alone in a sentence: **free morphemes**, also called unbounded morphemes, can stand alone as if they are words, **bound morphemes** can only be seen attached to a word. Morphemes can be further analysed in two categories, depending on whether they modify the grammatical role (i.e., part-of-speech) or the fundamental meaning of a word:

- **Derivational Morphemes** are bound morphemes that have the ability to modify the meaning and part-of-speech of a word (e.g.“boya” (the paint) - “boya-**cı**” (the painter)). Derivational morphemes are divided into 4 main categories:

- Morphemes attached to verbs to form nominals: (e.g., “sev-[2]” (to love) - “seve**cen**” (loving))
- Morphemes attached to verbs to form verbs (e.g., “böl-” (to split) - “böl**üş**-” (to share))
- Morphemes attached to nominals to form verbs (e.g., “kan” (blood) - “kan**a**-” (to bleed))
- Morphemes attached to nominals to form nominals (e.g., “çocuk” (child) - “çocuk**ça**” (childish))

- **Inflectional Morphemes** are bound morphemes attached to nouns, pronouns, nominal phrases, verbs, and verb frames that modify functional relations such as case, tense, voice, mood, person, and number. They are divided into 2 categories:
  - **Nominal inflectional morphemes** indicate number, possession, and case:
    - **Number Suffix:** The only number suffix is the plural suffix ‘lAr’[3] and used to indicate the plurality (e.g., “kedi**ler**” (cat**s**), “kitap**lar**” (book**s**))
    - **Possessive Suffixes:** They attribute possession to an object (e.g., “araba**m**” (**my** car), “araba**sı**” (**his/her/their** car))
    - **Case Suffixes:** They attribute the nominal to specific cases. In Turkish there are 6 case suffixes: absolute form (e.g., “okul” (school)), accusative (e.g., “okul**u**” (**the** school)), dative (e.g., “okul**a**” (**to** the school)), locative (e.g., “okul**da**” (**at** the school)), ablative (e.g., “okul**dan**” (**from** the school)) and genitive (e.g., “okul**un**” (**of** the school))
    - **Other Markers:** The other nominal inflectional markers are as follows: ‘*-(y)IA/ile*’ which can have comitative, instrumental or conjunctive meanings and can used as a bound or as a free morpheme (e.g., “Ahmet’**le**” or “Ahmet ile” (**with** Ali)), “*ki(n)*” that is used to form attributive adjectival phrases

[2]In Turkish, infinitives are given with - that either corresponds to -mak or -mek based on the vowel harmony with the root.

[3]In Turkish, capital letters in the suffixes indicate variability, I denotes ‘ı’ or ‘i’, A: ‘a’ or ‘e’, for example, *lAr* correspond to *ler* and *lar*.

(e.g., “okulda**ki** arkadaşlar” (the friends at the school)) and pronominal expressions (e.g., “okulda**ki**ler” (the ones at the school)).

– **Verbal inflectional morphemes** attach to verbs in two ways: finite verb forms, non-finite verb forms. A finite-verb acts as a verb in a sentence and a non-finite verb acts as a nominal in a sentence:

  * **Voice Suffixes:** Voice suffixes appear in finite and non-finite verb forms. They come immediately after the verb root and precede all other suffixes. They can be divided into 4 categories: causative (e.g., “kapa**t**-” (to close [s.t.])), passive (e.g., “bil**in**-” (to be known)), reflexive (e.g., “kurul**an**-” (to dry yourself)), and reciprocal suffixes (e.g., “gör**üş**-” (to see each other)).

  * **Negative Marker ‘-mA’:** They can appear in both finite and non-finite verb forms. They are located between the voice suffixes and the tense/aspect/modality markers (e.g., “anla**ma**-” (**not** to understand))

  * **Tense/Aspect/Modality Markers:** They can appear in both finite and non-finite verb forms. While **Tense** indicates the position of the state or action in time (e.g., “Geç**ti**n.” (You’ve passed.)), **Aspect** indicates the extension of the state or action over the time that can be perfective or imperfective (e.g., “Git**meli**sin.” (You should go.)), **Modality** indicates the reality of the state or action (e.g., “Kedi süt iç**er**.” (The cat drinks milk.)).

  * **Copula Markers:** Copula is a connecting word or a phrase in a particular form of a verb that links a subject and a complement. They appear only in finite verb forms. It is one of the peculiarities in Turkish grammar [54]. The copula in most cases corresponds to the verb “ol-” (to be) and has various forms: The zero copula, which is the third person rule, where two nouns or a single noun and an adjective are placed next to each other to make a sentence without a copula (e.g., “Öğretmenler üzgün.” (The teachers **are** sad.)), basic copula where sentences are created by the auxiliary verb ‘i-’ [55] (to be) by adding the auxiliary verb to a predicate, which can be nouns, adjectives or conjugated verb stems (e.g., “Öğretmenler üzgün**dür**.”

(The teachers **are** sad.)), negative copula refers to the negation word “değil” (not) (e.g., “Öğretmenler üzgün **değil**.” (The teachers are **not** sad.)), personal copula is used to create a complete sentence by adding a suffix to a noun or an adjective, as in a nonverbal person agreement (e.g., “(Onlar) Öğretmenler.” (They **are** the teachers.)), the past copula indicates the past form of the auxiliary verb “idi” (e.g., “Öğretmen**di**ler.“ (They **were** the teachers.)), the conditional copula indicates conditionality with the morpheme “ise” (e.g., “Öğretmenler**se**” (**If** they **are** the teachers))

∗ **Person markers:** They appear in finite verb forms and are attached to both verbal and nominal predicates to indicate the person of the subject (e.g., “Öğretmen değil**ler**di.” (**They** were not teachers.).

∗ **Subordinating Suffixes:** They appear only in non-finite verb forms and are the main means of forming subordinate clauses in Turkish. They are combined with verb stems to form nominals (e.g., “Okula **git-me-yeceğ-i** belli.” (It is clear **[that s/he won’t go to school]**.)).

3.1.2. Reduplication

Reduplication is used to repeat a word or part of a word, and it is performed in two forms in Turkish:

- **Emphatic Reduplication:** It is used to emphasize the quality of an adjective (e.g. “dar” (narrow) - “da**p**dar” (**very** narrow)).
- ***m*-Reduplication:** It is the generalization of the object to include also other similar objects, events, or states of affairs (e.g., “kapı **m**apı” (doors **and the like**)).
- **Doubling:** It is performed with the repetition of a word for nouns, adverbs, or adjectives. In this case, words can be used twice with and without *mI*[4] (e.g., “yavaş yavaş” (slowly), “sarı sarı (yapraklar)” (many yellow (leaves)), “poşet poşet

[4]*mI* corresponds to the interrogative morpheme.

(kıyafetler)” (many bags of (clothes)), “sakin mi sakin bir deniz” (a remarkably calm sea)) or with similar sounding words (e.g., “ufak tefek” (tiny), “çoluk çocuk” (wife and children)).

3.1.3. Clitics

Clitics are meaning-bearing entities that behave like independent words although they are considered affixes at the morphological level [56]. The clitics in Turkish are as follows [57]:

- **“mI”** has the following meanings:
  - Question particle in Yes/No questions (e.g., “İster **miyiz**?” (**Do** we want?))
  - Adverbial clause marker (e.g., “Geldin **mi**” (**As soon as** you come))
  - Intensifier in doubled forms (e.g., “ilginç **mi** ilginç” (very interesting))
- **“dA”** has the following meanings:
  - Additive (e.g., “Görmedim **de**.” (**Moreover**, I didn’t see it.))
  - Adversative (e.g., “Görmedim **de** gösterdiler.” (I didn’t see it **but** people showed it to me.))
  - Continuative/Topic Shifting (e.g., “Bu kitabı **da** Ahmet getirdi.” (**As** for this book, Ahmet brought it.))
  - Enumerating (e.g., “Ahmet **de** Ali **de**” (**Both** Ahmet **and** Ali))
- **“-(y)sA/ise”** has the following functions:
  - Topic Shifting (e.g., “Akşam**sa** tiyatroya gittik.” (In the evening, we went to theatre.))
  - Contrastive (e.g., “Ahmet **ise**” (**As** for Ahmet))
- **“ya”** has the following meanings:

  - Contractive/Adversative Conjunction (e.g., “İzledim **ya** anlamadım.” (I watched it **but** I couldn’t understand it.))
  - Repudiative/Discourse Connective (e.g., “İzlemedim dedim **ya**!” (I told you that I didn’t watched it!))
  - Stressable discourse connective (e.g., “**Ya** gelemediyse?” (**What if** s/he couldn’t come?))

- **“ki”** has the following meanings:
  - Subordinate connection to a noun clause (e.g., “İnanıyorum [**ki** onlar gelecek].” (I believe [**that** they will come].))
  - Subordinate connection to an adverbial clause (e.g., “Not alıyorum [**ki** sonradan sana hatırlatayım].” (I am taking notes [**so that** I’ll remind you later].))
  - Repudiative Discourse Connective (e.g., “Gidemiyorum **ki**.” (I **just** can’t go.))
  - Exclamations (e.g., “İstanbul yazın o kadar sıcak olur **ki**!” (İstanbul is so hot in summer!))
  - Relative Clause Marker (e.g., “Bugün okula gelirse, [**ki** geleceğini hiç sanmıyorum,] oyuna gidecekmiş.” (If s/he will come to school today, **which** I don’t think s/he will, s/he will go to the game.))
  - Non-Restrictive Relative Clauses (e.g., “Ahmet [**ki** çok çalışkan], o bile sınavdan düşük not aldı.” (Even Ahmet, [**who** is very hardworking] got a low grade in the exam.))
  - Restrictive Relative Clauses (e.g., “Bir yazar [**ki** dilekçe yazmayı bilmesin], onun kitabı okunulmz.” (A writer [**who** doesn’t know how to write a petition], isn’t to be trusted to read her/his book.))

- **“bile”** has the following meaning:
  - Additive Connective (e.g., “İzledim **bile**.” (I have **already** watched it.))

3.2. Word Classes

A word in Turkish can be assigned to one of the following syntactic categories: nominal (i.e., noun, pronoun, adjective, or adverb), verb, postposition, conjunction or discourse connective, and interjection [54].

1. **Nominal** contains four categories:

    (a) **Noun** is a word for a thing (e.g., “elma” (apple)), a person (e.g., “adam” (man)), an abstract concept (e.g., “hüzün” (sadness)) or a proper noun of a thing (e.g., “UNESCO”), person (e.g., “Ayşe”) or a place (e.g., “İstanbul”).

    (b) **Pronoun** is a word that is used as a substitute for a noun or noun phrase that refers to the participants in the given context (e.g., “sen” (you), “ben” (I), “bu” (this), “başkası” (another), “şurada” (here)).

    (c) **Adjective** describes a noun by quality, property or status (e.g., “yumuşak” (soft), “önemli” (important), “kırmızı” (red)). Determiners (e.g., “bir” (a/an), “her” (each), “bütün” (all)) and numerals (e.g., “birinci” (the first), “üç yüz” (three hundred)) also belong to this class.

    (d) **Adverb** is used to specify a verb, an adjective, another adverb, or an entire sentence. It expresses verbs by providing information about the manner (e.g., “yavaşça (slowly)), time (e.g., “şimdi” (now)), or degree of occurrence of an event (e.g., “hep” (always)). Adverbs that describe adjectives or other adverbs provide the degree to which the concepts they denote apply (e.g., “çok” (), “çok fazla” (too much), “daha fazla” (more)). Examples of adverbs that refer to a sentence are “muhtemelen” (possibly) and “maalesef” (unfortunately).

2. **Verb** is used to describe an action, event, process, state, or occurrence (e.g., “gel-” (to come), “bisiklet sür-” (to ride), “bit-” (to end), “ol-” (to be), “kal-” (to remain)).

3. **Postposition** is a word that takes a noun, pronoun, or noun phrase and indicates its relation to another word in the sentence (e.g., “karşı” (against), “sonra” (after), “için” (for)).

4. **Conjunction and Discourse Connective** Conjunction joins two or more sentences, clauses, or words (“de $\cdots$ de” (both $\cdots$ and), “fakat” (but), “ve” (and)) (e.g., “kitap ve defter” (book and notebook), “Ali de Ahmet de” (both Ali and Ahmet)). Discourse connective is used within a sentence or sentences to provide cohesion in discourse. They can introduce statements that are a development of a previous statement (e.g., “hatta” (moreover), “üstelik” (furthermore)), or they can be used to expand on a previous statement (e.g., “yani/başka bir deyişle” (in other words)), or to state a fact that seems to contradict what was just said (e.g., “halbuki” (whereas)).

5. **Interjection** is a word or a phrase that expresses feelings (e.g., “wow” (wow), “hay allah” (oh dear)) or the speaker’s attitude towards the hearer (e.g., “yahu” (hey)) to initiate a conversation.

3.3. Syntax

3.3.1. Constituents of a Sentence

In Turkish, sentences can be simple or complex in terms of the subordinate clauses that are semantically dependent on the main clause[5].

Examples:

- Tiyatroya gittik. (in English, *“We went to the theatre.”*)
- Dün [bulaşıkları yıkarken], [yıllardır görmediğim] arkadaşım aradı. (in English, *“Yesterday, [as I was washing the dishes], my friend [whom I hadn’t seen for years] called.”*)

Two main components of a simple sentence are the subject and the predicate. In a sentence, the **predicate** expresses an event, process, or state of affairs with an agreement of the subject,

[5]Subordinate clauses are indicated by [] as defined in [57]

and the **subject** of the sentence may be a person, place, or thing that possesses or is affected by the predicate:

Examples:

- **Kerem** bir an **durdu**. (in English, *"**Kerem stopped** for a moment."*)
- **O** öğretmen**di**. (in English, *"**S/he is** a teacher."*)
- **İstanbul**, Türkiye'nin en büyük şehri**dir** ve Avrupa ve Asya'yı birbirine **bağlar**. (in English, *"**Istanbul is** Turkey's the largest city and **connects** Europe and Asia."*)

Turkish is a syntactically free order language. However, the unmarked order (neutral word order) is subject-(object)-predicate (SOV) (e.g. "Ahmet okula gitti." (in English, *"Ahmet went to the school."*)).

The sentence types are divided into 2 classes based on the predicate:

1. Verbal sentences have predicates that are finite verbs:

   Examples:

   - Kerem bana **baktı**. (in English, *"Kerem **looked at** me."*)
   - Bugün okula **gideceğim**. (in English, *"I will **go** to school today."*)

2. Nominal sentences have predicates that do not contain a verb or use a verb in the form of the copula (e.g., ol- (be)):

   Examples:

   - Bir tutsağ**ım** ben. (in English, *"**I am** a prisoner."*)
   - Ahmet doktor **ol**acaktı. (in English, *"Ahmet **was** going to **be** a doctor."*)

3.3.2. Phrases

A phrase is a syntactic unit, which is a collection of syntactically coherent words that functions as a unit in a sentence. There are three types of phrases: verb phrase, noun phrase, and prepositional phrase.

3.3.2.1. Verb Phrase A verb phrase is composed of a verb, and additionally may contain a complement of a verb or an adverbial:

Examples:

- **Yürüyorum**. (in English, *"**I am walking**."*)
- **Hızlı koşma**! (in English, *"**Do not run fast!**"*)
- Ahmet **suyu soğuk soğuk içti**. (in English, *"Ahmet **drank the cold water too fast**."*)

3.3.2.2. Noun Phrase A noun phrase primarily contains a noun or a group of words that contain a noun. The role of a noun phrase is that of a subject and some kind of complement such as object, subject complement, and complement of a postposition:

Examples:

- **Ahmet** geldi. (in English, *"**Ahmet** came."*)
- **Yavru kedi** çok tatlıydı. (in English, *"**The kitten** was so cute."*)
- Onlar **yeni öğretmenler**. (in English, *"They are **the new teachers**."*)
- **[Ahmet'in sevdiği] ekşi elma** bitmişti. (in English, *"**[Ahmet's favorite] sour apple** was finished."*)

3.3.2.3. Postpositional Phrase

A postpositional phrase consists of a noun phrase and a postposition that follows the noun phrase. While the postposition is the head of the phrase, the noun phrase is the complement of the phrase:

Examples:

- **Akşama doğru** yağmur yağabilir. (in English, *"It may start raining **towards evening**."*)
- **İki günden beri** okula gitmiyor. (in English, *"S/he has not been to school **for two days**."*)
- **Her oyundan önce** bir bardak çay içer. (in English, *"S/he drinks a cup of tea **before every game**."*)

3.3.3. Complex Sentences and Subordination

Complex sentences consist of at least one subordinate clause in addition to the main clause, where a clause is a group of words that functions as a unit in a sentence. In Turkish, there are three types of clauses: noun, relative, and adverbial clause.

3.3.3.1. Noun Clause

It occurs as a noun phrase in a complex sentence as the subject or object in the sentence:

Examples:

- Ahmet [senin okula gittiğini] bilmiyordu. (in English, *"Ahmet didn't know that [you were going to school]."*)
- Herkes [kitap yazmanı] istiyor. (in English, *"Everyone wants [you to write a book]."*)

3.3.3.2. Relative Clause

It occurs as an adjectival phrase that is used to modify noun phrases:

Examples:

- [Öğretmen olan] oğlu, Ankara'da yaşıyor. (in English, *"Her/His son, [who is a teacher], lives in Ankara."*)
- [Evimizde beslediğimiz] kedi dün hastalandı. (in English, *"The cat [we feed at home] got sick yesterday."*)

3.3.3.3. Adverbial Clause

It occurs as an adverbial phrase that expresses the verb in terms of time, place, manner, and degree:

Examples:

- [Kedimiz üşümesin diye] ısıtıcıyı açtık. (in English, *"We turned on the heater [so that our cat would not get cold]."*)
- Tam dışarı çıkacaktık [ki kapı çaldı]. (in English, *"We were about to go out [when the doorbell rang]."*)
- [İlham geldi mi] şarkı söylemeye başlar. (in English, *"[When inspired], she starts to sing."*)
- [Birazdan yağmur yağacağı] söyleniyor. (in English, *"It is said [that it will rain soon."*)

3.3.4. Conditional Sentences

Conditional sentences are based on a condition that is met when a certain action takes place:

3.3.4.1. Predictive Conditionals It expresses a predictable relation between two situations:

Examples:

- [Okula gitmezsen] sınıfta kalacaksın. (in English, *"[If you do not go to school] you will fail the class."*)
- [Çiçekler sulanmazsa] solacaklar. (in English, *"[If the flowers are not watered] they will wither."*)
- [Evini satacak olursan] ben almak isterim. (in English, *"[If you sell your house] I would like to buy it."*)

3.3.4.2. Knowable Conditionals It serves as a background for an inference, question, or some kind of volitional utterance:

Examples:

- Ahmet [[Ali'nin geldiğini] biliyorsa] yemeğe gelmeyecektir. (in English, *"[If Ahmet knew [that Ali came]] he will not come to dinner."*)
- [Yemek hazırsa] hızlıca yiyebiliriz. (in English, *"[If the food is ready] we can eat quickly."*)
- [Sınavlara girse bile] mezun olamaz. (in English, *"[Even if s/he takes the exams] s/he will not graduate."*)

3.3.4.3. Universal Conditional Clauses It contains a question phrase that corresponds to a "-ever" clause such as *whoever* and *wherever* in addition to a conditional marker:

Examples:

- Ahmet [ne zaman maçta oynasa] takım her zaman galip olur. (in English, *"[Whenever Ahmet plays in the game] the team wins."*)
- [Kime sorduysak] hepsinden aynı cevabı aldık. (in English, *"[Whoever we asked] we had the same answer."*)
- [Ne kadar iyi olsan da] o okuldan kabul alamazsın. (in English, *"[However good you are] you cannot get an offer from that school."*)

4. SEMANTIC PARSING MODELS FOR THE UCCA REPRESENTATION

Semantic parsing is the task of mapping a text given in a natural language to its formal representation, which provides an abstraction of its meaning that can be easily processed by a machine to serve a particular NLP task.

In this chapter, we consider UCCA parsing, a semantic parsing in which the semantic structure is the UCCA graph. UCCA parsing is challenging compared to other syntactic parsing problems such as dependency parsing or constituency parsing. First, UCCA is represented by graphs rather than trees as in syntactic parsing. Moreover, the UCCA representation allows reentrancy with remote edges to support argument sharing, resulting in an acyclic graph, non-terminal nodes compromising several tokens that are jointly considered as a single entity according to semantic or cognitive consideration, and discontinuity corresponding to non-projectivity. These features motivate the development of supervised and unsupervised parsing models for UCCA parsing.

In this chapter, we present the proposed models, which are deep neural network models, one of which is unsupervised, and attempt to answer the following questions:

Research Questions:

1. Which approach (transition-based, graph-based) is most effective for the UCCA-based semantic parsing problem?
2. Are the constituency parsing models applicable to the semantic parsing problems after preprocessing the UCCA representation?
3. Is the transition set of the transition-based parser important to learn the features of the UCCA representation (discontinuous units, reentrancy, etc.)?
4. Given the problem of data scarcity in UCCA representation, up to what extent does an unsupervised model learn semantic parsing?

4.1. Related Work

With the published UCCA framework and datasets, the TUPA parser [46] is proposed, which is the first parser to generate UCCA representations. It is a neural transition-based parser model with additional transitions to handle discontinuous and remote nodes in UCCA graphs. The transition `NODE` creates new non-terminal nodes, `LEFT-EDGE` and `RIGHT-EDGE` create a new primary edge between the first two elements on the stack whose parent is the left and right nodes, respectively. `LEFT-REMOTE` and `RIGHT-REMOTE` create a new remote edge between the first two elements on the stack, creating reentrancy and the form of DAG. Finally, `SWAP` pops the node on the stack and adds it to the top of the buffer, creating a discontinuous unit in the UCCA representation.

In addition to the commonly used features (word, PoS tag, syntactic dependency label of the top four stack elements, and the next three buffer elements), the features used in the study are those related to discontinuous nodes [58], the number of remote children, which is a UCCA specific feature, features for existing edges, the number of parents and children. Since the TUPA parser [46] is the first parser for the UCCA representation, the authors also investigate the impact of the transition classifier with three different models: TUPA$_{Sparse}$ with a linear classifier with sparse features trained with the averaged structured perceptron algorithm [59] MINUPDATE [60], TUPA$_{MLP}$ using a Multi-Layer Perceptron (MLP) with two linear layers, and TUPA$_{BiLSTM}$ using BiLSTM model, where the output of BiLSTM with dense features representing the parser state (e.g., existing edge labels and previous parser actions), is concatenated and fed into a feed-forward network similar to TUPA$_{MLP}$ to select the next transition. The architecture of the TUPA$_{BiLSTM}$ is given in Figure 4.1.

Hershcovich et al. [61] extend the TUPA parser with multi-task learning using other semantic graph representations, namely AMR [15], UD [62, 63] and SDP [36]. In multi-task transition-based parsing, TUPA$_{BiLSTM}$ was used for UCCA parsing, the main task, and a BiLSTM was added as a shared BiLSTM for all tasks with replication of the MLP for each task. The architecture of the MTL model is given in Figure 4.2. In order to be able to apply the parser to all semantic graph representations (UCCA, AMR, UD, and SDP),

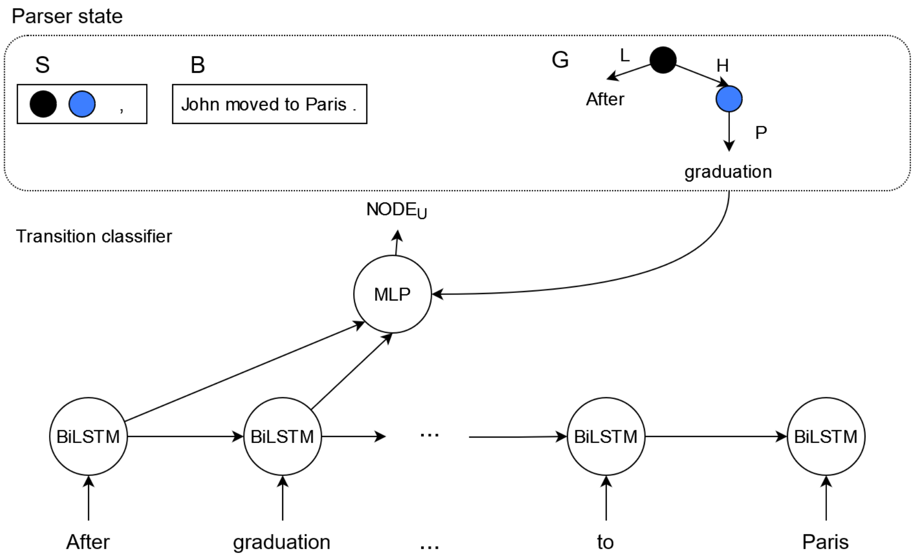


Figure 4.1 The architecture of the TUPA$_{BiLSTM}$ parser

the representations were converted to a unified DAG format that allows all formats to be represented with very little loss.

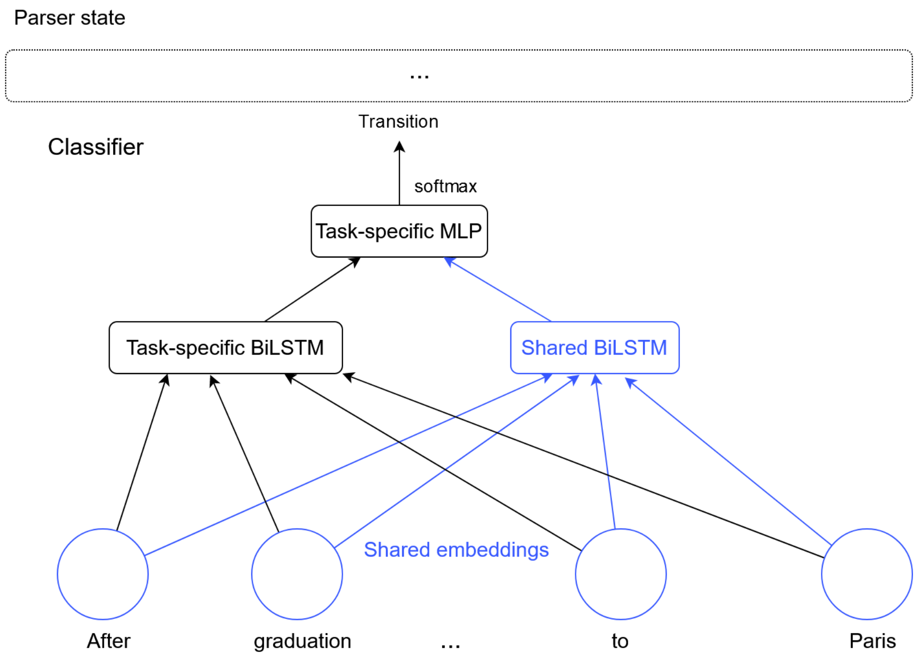


Figure 4.2 The architecture of the MTL model

The UCCA framework has been the main theme in some share tasks, e.g., “Cross-lingual Semantic Parsing with UCCA” at SemEval 2019 [1] and “Meaning Representation Parsing

(MRP)" cross-framework shared task [40, 64] in 2019 and 2020.

Current UCCA-based semantic parsers can be categorized based on their approaches as follows:

- **Transition-based [46, 65–67]**: Transition-based approaches define a sequence of actions that eventually build semantic graphs. Since the first proposed model is the TUPA parser [46], transition-based approaches extend the TUPA parser by adding extra transitions [68, 69], new features [65, 70], or layers [66, 71].

  Arviv et al. [68] add `LABEL`, `PROPERTY` and `ATTRIBUTE` transitions with an additional classifier evoked with a set of label/property/attribute values due to the large number of node labels and properties in the MRP frameworks (EDS, PTG, UCCA, AMR, DRG). Lai et al. [69] add the transition `RESOLVE` to the TUPA parser to deal with multiple nodes and their dependencies, which is a very complex problem for the transition `REDUCE`.

  Pütz and Glocker [65] implement the TUPA parser with a set of features based on the top three items on the stack and the buffer, as well as deep contextualized word embeddings of the rightmost and leftmost parents and children of the respective items. The authors also use additional training data for English from the 1B-word benchmark [72] and for German from the archive of the newspaper taz. Bai and Zhao [70] extend the TUPA parser with hand-crafted features that are existing node labels related to the top four stack elements and the first three buffer elements, as well as the last three actions performed by the parser, and, when there are fewer than three actions before, zero embeddings instead. These features are mapped to embeddings that are trained when the model is trained.

  Dou et al. [67] use Stack-LSTM instead of BiLSTM of TUPA parser for UCCA parsing. Lyu et al. [66] use TUPA$_{BiLSTM}$ and TUPA$_{MLP}$ as a cascaded parser with a multi-stage training procedure, first training TUPA$_{BiLSTM}$ and retraining the model with TUPA$_{MLP}$.

- **Graph-based [73–80]**: Graph-based approaches generally approach the task as a search problem of finding the graph with the highest score among all possible graphs for a given input. Some of the approaches leverage existing neural parser architectures introduced specifically for dependency parsing (NeurboParser [81], JAMR [82] and UDPipe [83]), while others introduce neural architectures for UCCA-based semantic representation using a graph-based approach [73, 76, 77, 80].

  The most common studies with comparable results apply constituency parsing models [73, 74, 77, 80] to the UCCA representation after preprocessing that converts DAGs into constituency trees. Jiang et al. [73] solve the semantic parsing problem for UCCA by converting the UCCA representations into constituency trees by removing remote edges and adding a `ROOT` node and applying the minimal span-based parser of Stern et al. [84] with an additional classifier for remote edges. Li et al.[77] and Zhang et al. [80] directly follow the graph-based UCCA parser of Jiang et al. [73]. Cao et al. [74] apply the minimal span-based parser of Stern et al. [84] with a different conversion method UCCA representations into constituency trees. The authors added some auxiliary nodes `TOP`, `HEAD` for non-terminal nodes and `TOKEN` and `MWE` terminal nodes.

  Koreeda et al. [76] propose an encoder-biaffine architecture for all representations presented in MRP 2019 [40] that uses a shared encoder and biaffine network to predict edges with a pointer network to generate non-terminal nodes of the UCCA representation and an additional BiLSTM to encode the context of the terminal and non-terminal nodes.

- **Composition-based [85]**: Composition-based approaches follow the compositionality principle and perform semantic parsing as the result of a derivation process that incorporates both lexical and syntactic-semantic rules to develop a semantic graph parser.

  Donatelli et al. [85] apply the AM dependency parser of Lindemann et al. [86] after converting UCCA annotations into AM dependency graphs.

- **Encoder-decoder-based [78, 87, 88]**: Encoder-decoder approaches use an encoder-decoder architecture to convert an input sentence into a semantic graph as performed in NMT.

  Na et al. [78] present an encoder-decoder architecture for the parsing problem, in which BERT-BiLSTM is the encoder and the biaffine attention is the decoder. To process the UCCA representation, it is converted to a bilexical framework using the `semstr` tool before training the model, and then converted back to UCCA format.

  Yu and Sagae [87] follow an encoder-decoder architecture using BiLSTM as the encoder and self-attention as the decoder, as used by Kitaev and Klein [89]. For non-terminal nodes with more than one terminal, the span representation of Cross and Huang [90] is applied before the decoder of the model.

  Ozaki et al. [88] apply Plain Graph Notation (PGN) as a universal notation with a number of pseudo-nodes indicating the end of node prediction, end of label prediction, etc. for the semantic frameworks presented in MRP 2020 [64]. The authors present an encoder-decoder architecture where the pre-trained contextualized embedding BERT [91] is the encoder and Transformer [92] is the decoder of the architecture that generates PNG that is transformed into a semantic representation.

4.2. Proposed Models

In this section we describe the proposed models, which are supervised and unsupervised models. The supervised models are self-attentive and incremental semantic parsers that are graph-based and transition-based, respectively. The unsupervised model is the chart-based zero-shot parser, which is a fully unsupervised model.

4.2.1. Graph-based Semantic Parser (Self-Attentive Semantic Parser)

We adopt the constituency parsing model based on the self-attention mechanism proposed by Kitaev and Klein [93] to learn the UCCA semantic representations of a given text. The parser is built on an encoder-decoder architecture, where the encoder is based on self-attention mechanism and the decoder is based on the CYK (Cocke-Younger-Kasami) algorithm [94]. The overall view of the encoder-decoder architecture is given in Figure 4.3. The parser follows a chart-based constituency parsing approach where the constituency tree T of an input sentence $s = \{w_1, \cdots, w_n\}$ with words w_i is defined as a set of labeled spans:

$$T = \{(i_t, j_t, l_t) : t = 1, \cdots, |T|\} \tag{3}$$

where i_t and j_t refer to the beginning and ending positions of the t^{th} span respectively with the label set $l_t \in L$.

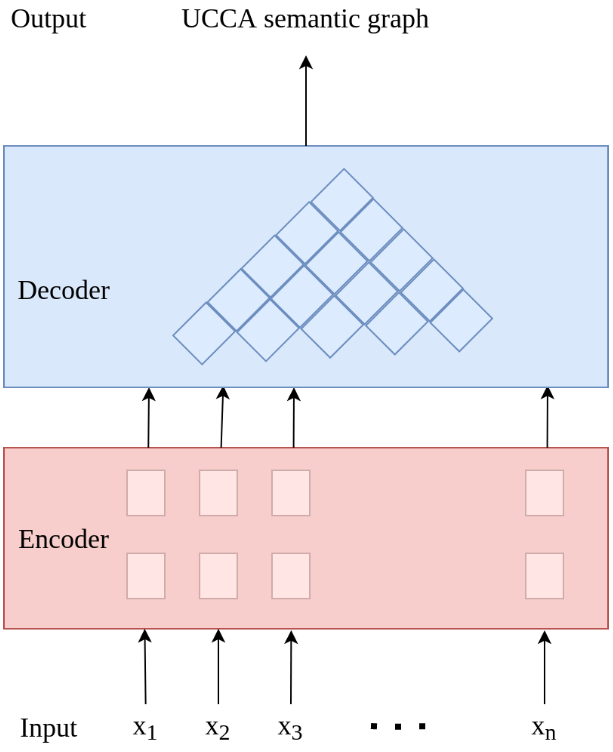


Figure 4.3 The overview of the self-attentive semantic parser model

We assign a score $s(T)$ to each tree, which is decomposed as follows:

$$s(T) = \sum_{(i,j,l) \in T} s(i, j, l) \tag{4}$$

Here, $s(i, j, l)$ denotes per-span scores predicted by the model.

Each word w_t is mapped into a dense vector x_t which is a concatenation of the word embedding e_{w_t}, PoS tag embedding e_{p_t}, dependency label embedding e_{d_t}, entity type embedding e_{e_t}, and entity iob (inside-outside-beginning) category embedding $e_{e_ob_t}$:

$$x_t = e_{w_t} \oplus e_{p_t} \oplus e_{d_t} \oplus e_{e_t} \oplus e_{e_ob_t} \tag{5}$$

The overview of the encoder along with the remote edge recovery is given in Figure 4.4. The encoder consists of multiple self-attention layers[6] and only one of them is depicted in the figure for simplicity reasons. The encoder learns a context vector y_t for each position t for a word vector x_t.

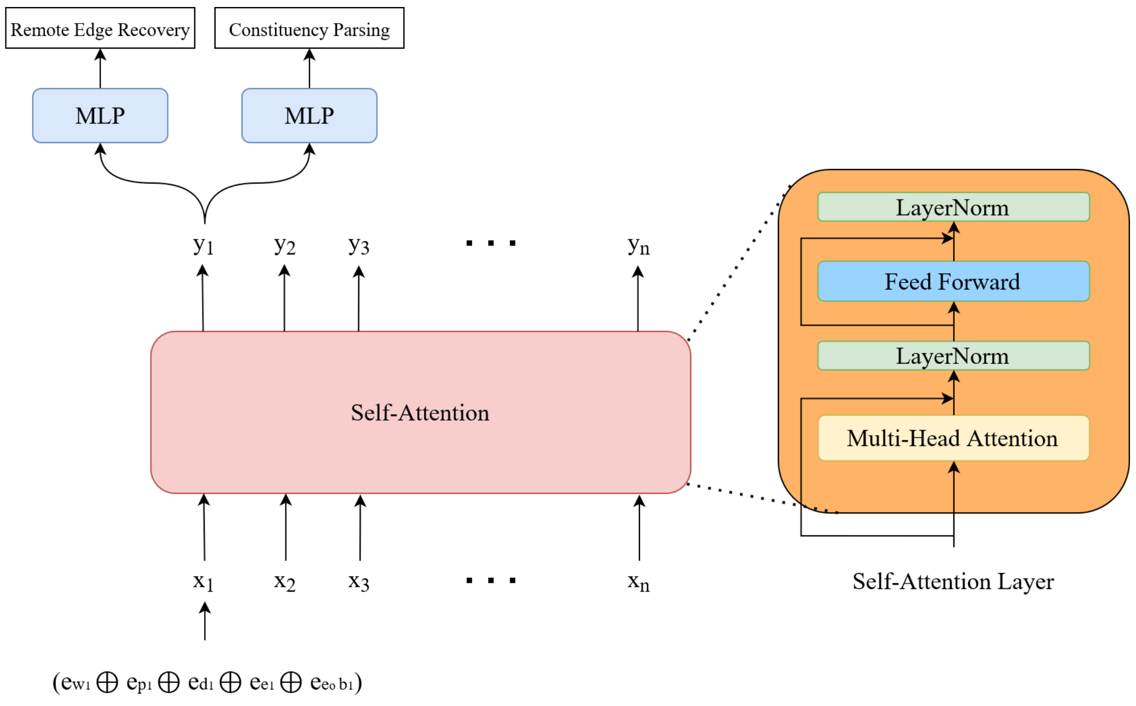


Figure 4.4 An overview of the self-attention encoder

An MLP classifier with two fully-connected layers with ReLU activation function assigns labeling scores $s(i, j, l)$ to each span using the encoder output. We integrate remote edge recovery that also shares the same encoder to recover remote edges in trees [73] as shown

[6]In our model, the encoder involves 8 self-attention layers.

in Figure 4.4. Therefore, the model incorporates two independent MLPs to predict remote edges and candidate parent nodes that use the same encoder.

The parsing loss is the sum of the cross-entropy losses introduced by both remote edges L_{remote} and non-terminal node pairs $L_{non-terminal}$ as indicated below:

$$L = L_{remote} + L_{non-terminal} \tag{6}$$

As for inference, CYK (Cocke-Younger-Kasami) algorithm [94] is used to generate a globally optimized tree $\hat{T}$ for each sentence that acts as a decoder in the model (see Figure 4.3). Therefore the tree with the maximum score is identified by the CYK algorithm as follows:

$$\hat{T} = \arg\max_{T} s(T) \tag{7}$$

4.2.2. Transition-based Semantic Parser (Incremental Semantic Parser)

We adopt the attach-juxtapose transition system of Yang and Deng [95] for the semantic parsing problem. We use the same architecture of the TUPA parser [46] to compare the results of the same architecture with a different set of transitions for the semantic parsing.

The transition-based constituency parsing model called `attach-juxtapose` [95] allows strongly incremental parsing with two transitions `attach` and `juxtapose`. For a sentence $s = w_1, \cdots, w_n$, it starts with an empty tree and performs n actions that integrate the tokens from the buffer containing words of the sentence into the tree. While the transition `attach` attaches the new token as a child to an existing node in the tree, `juxtapose` juxtaposes the new token as a sibling to an existing node in the tree by creating a shared parent node.

4.2.2.1. Attach-juxtapose transition system for UCCA parsing We extend the transition set of `attach-juxtapose` [95] for the TUPA parser [46], which is a transition-based parser for the UCCA representation. The architecture of the TUPA parser [46] with updated transition set is given in Figure 4.5.

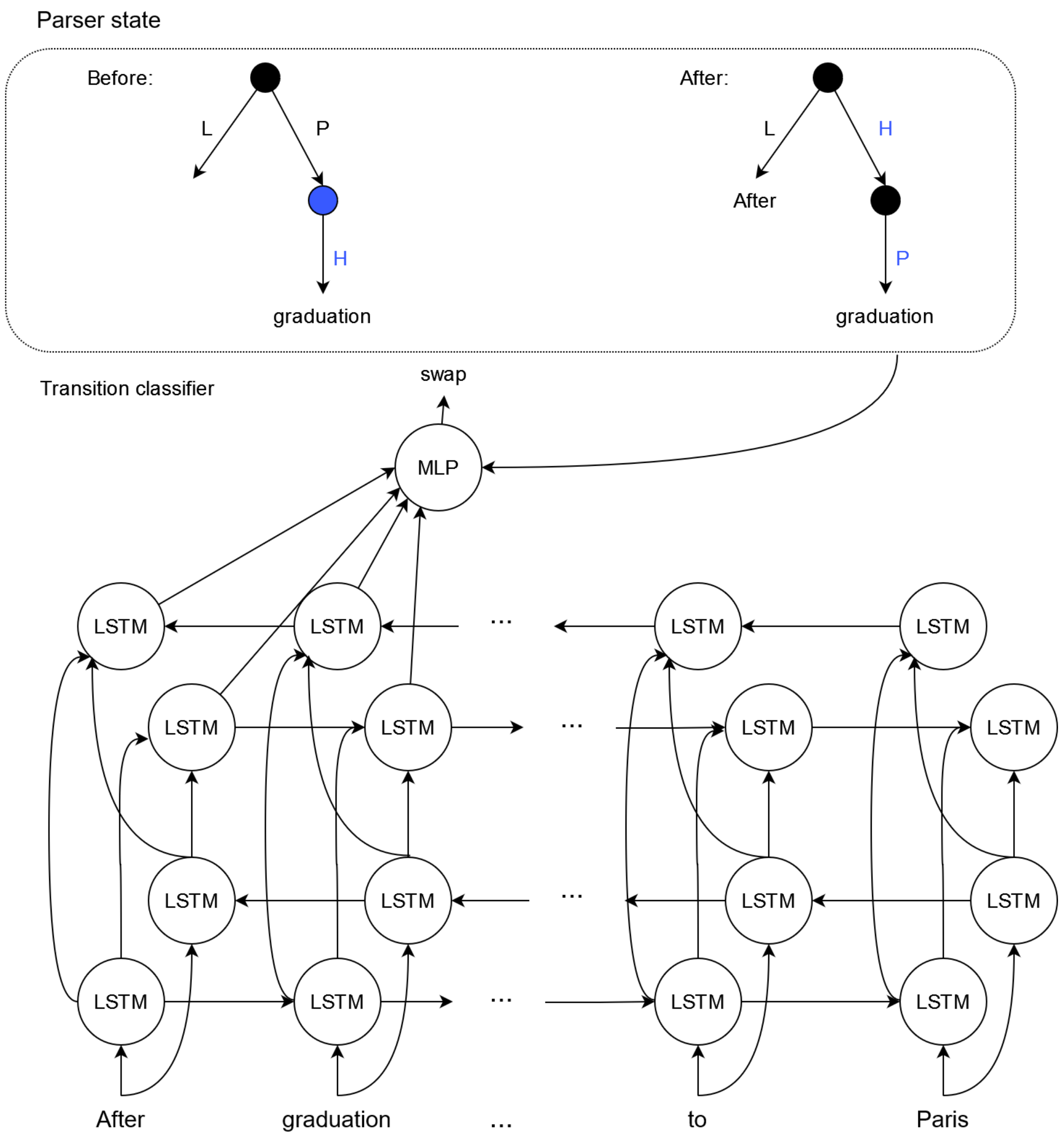


Figure 4.5 Architecture of the TUPA model extended by attach-juxtapose transition set

Transition Set Given a sentence $s = w_1, \cdots, w_n$, we predict a UCCA graph G over the sequence using 3 transitions. In addition to the standard transitions, `attach` and `juxtapose`, used in the `attach-juxtapose` [95] model, we add `swap` as the TUPA parser.

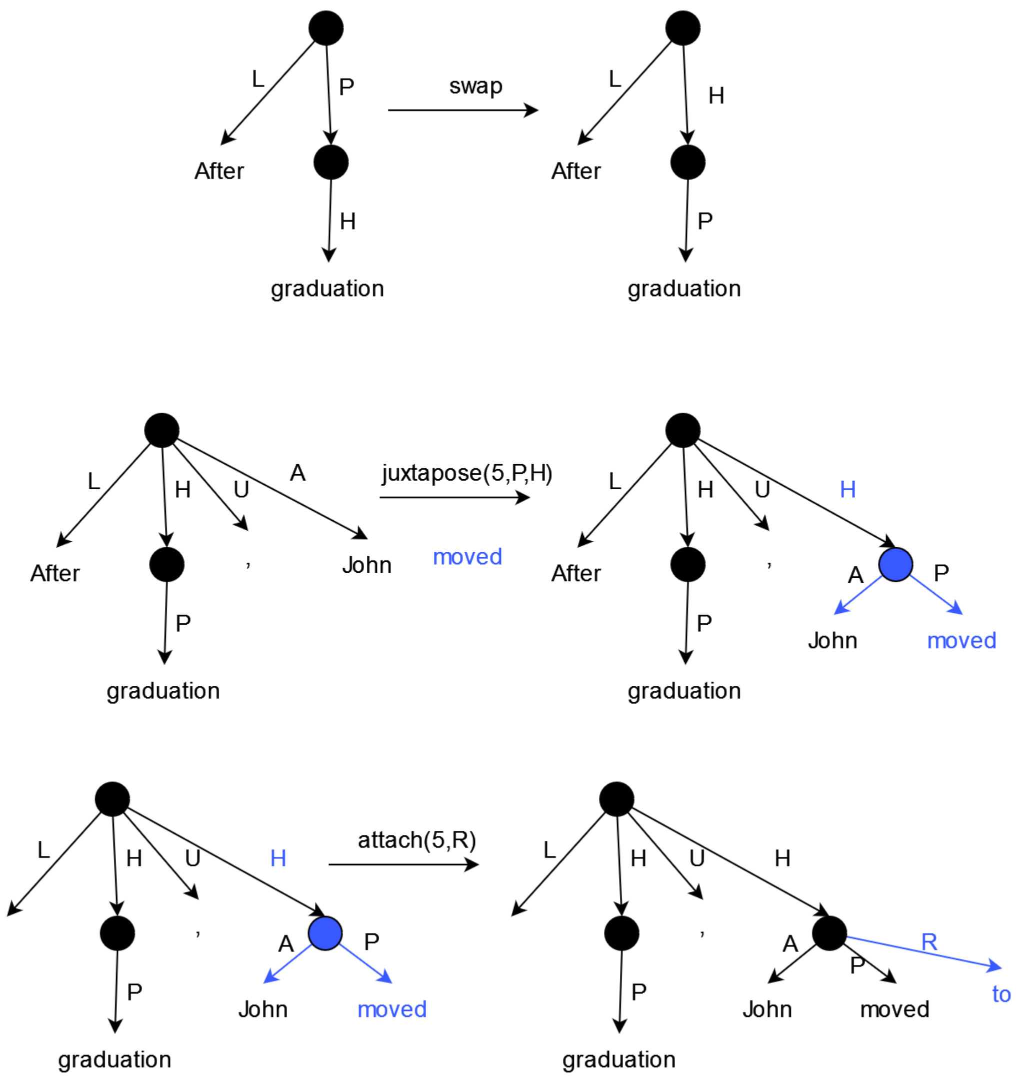
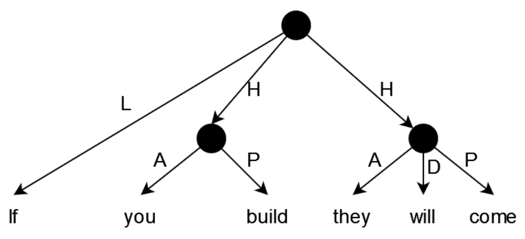


Figure 4.6 Transition set in the incremental parser

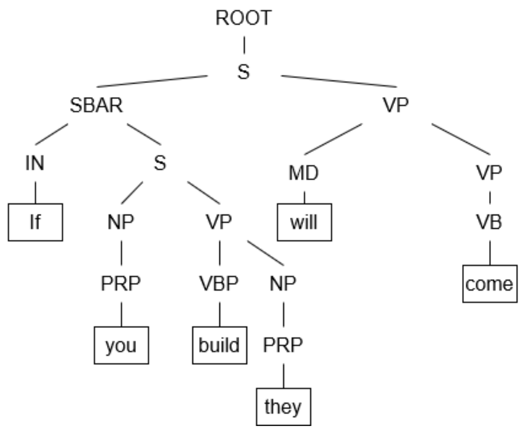
- `attach(target_node, parent_label)`: Attach the token as a descendent of target_node with given parent_label. If the target_node is not a non-terminal node, a new node is created before attaching the token to target_node.

- `juxtapose(target_node, parent_label, new_label)`: Create an internal node with the given label as the shared parent of the target_node and the new token. It then replaces target_node in the tree.

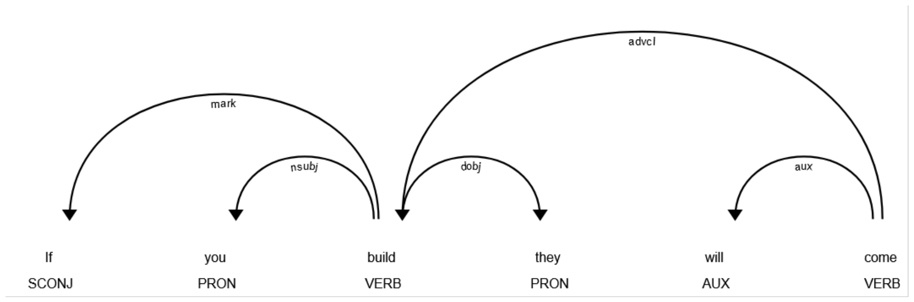
- `swap`: Pop the second node on the stack and add it to the top of the buffer.



(a) Semantic Parsing



(b) Constituency Parsing



(c) Dependency Parsing

Figure 4.7 Representation of sentence *"If you build they will come"* in 3 representations

4.2.3. Chart-based Zero-shot Semantic Parser

We exploit the concept of syntactic distance [96, 97], which has been explored in particular for constituency parsing [98–100] by using the PLMs directly without fine-tuning. Here, we adopt chart-based zero-shot parsing based on the syntactic distance for three different parsing problems, namely semantic, constituency, and dependency parsing, to explore the usability of PLMs in the zero-shot setting.

An example of a UCCA representation, constituency tree, and dependency tree for the sentence *"If you build they will come"* is shown in Figure 4.7.

The method calculates scores for spans where an input sentence $s = \{w_1, \cdots, w_n\}$ is made up of a set of labeled spans as follows:

$$T = \{(i_t, j_t, l_t) : t = 1, \cdots, |T|\}$$

where i_t and j_t refer to the beginning and ending positions of the t^{th} span respectively with the label set $l_t \in L$. A score $s(t)$ is assigned to each tree, which is decomposed as follows:

$$s(t) = \sum_{(i,j)\in t} s_{span}(i, j) \tag{8}$$

Here, $s_{span}(i, j)$ denotes per-span scores that are calculated recursively by splitting spans into smaller spans as defined below:

$$\begin{aligned} s_{split}(i, k, j) &= s_{span}(i, k) + s_{span}(k + 1, j) \\ s_{span}(i, j) &= s_{comp}(i, j) + min_{i\le k<j} s_{split}(i, k, j) \end{aligned} \tag{9}$$

where $s_{comp}(\cdot, \cdot, \cdot)$ measures the validity of the compositionality of the $span(i, j)$ itself and $s_{split}(i, k, j)$ measures the scores for how possible to split span (i, j) at position k. To calculate $s_{comp}(\cdot, \cdot, \cdot)$, [99] introduced two alternative labeled functions: $s_c(\cdot, \cdot)$ characteristic score function and $s_p(\cdot, \cdot)$ pair score function. The pair score function computes the average pairwise distance in a given span:

$$\begin{aligned} s_p(i, j) &= \frac{1}{\binom{j - i + 1}{2}} \sum_{(w_x, w_y)\in pair(i,j)} f(g(w_x), g(w_y)) \\ s_c(i, j) &= \frac{1}{j - i + 1} \sum_{i\le x\le j} f(g(w_x), c) \\ c &= \frac{1}{j - i + 1} \sum_{i\le y\le j} g(w_y) \end{aligned} \tag{10}$$

where $pair(i, j)$ calculates all combinations of bigrams (e.g. w_x, w_y) inside the span (i, j), $f(\cdot, \cdot)$ is a distance measure function (Jensen-Shannon (JSD) and Hellinger (HEL)) that

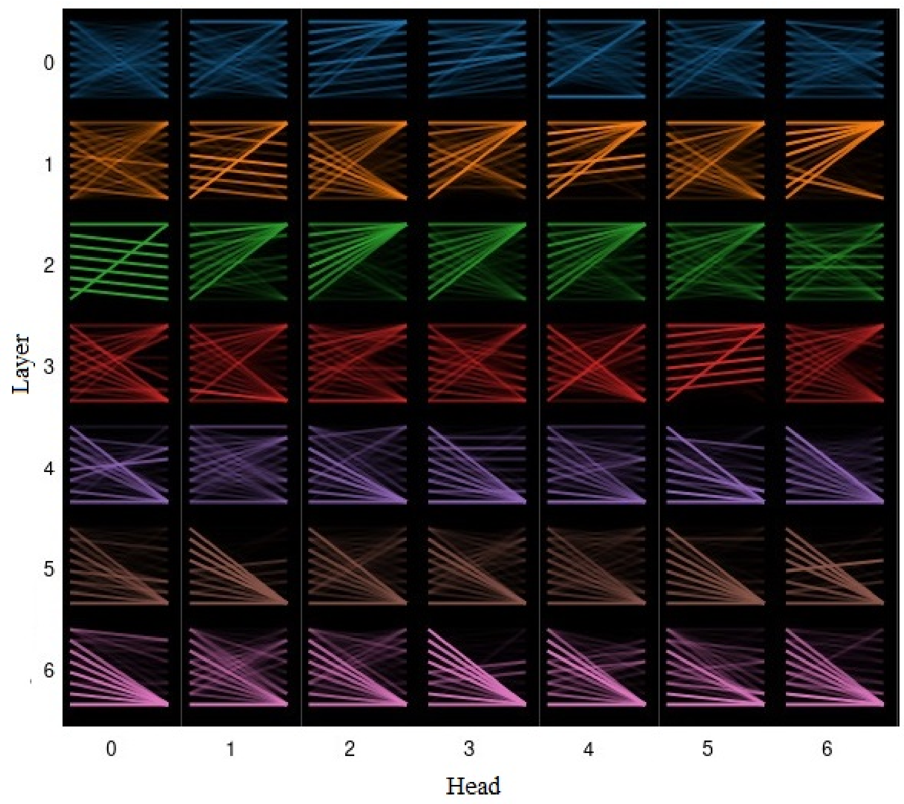


Figure 4.8 Model view (first 7 layers and 7 attention heads) for input sentence *"The cat sat on the mat"*

measures the distance between two spans, and $g(\cdot)$ is the representation function that yields the v^{th} attention head on the u^{th} layer of the pre-trained language model for $g = \{g^d_{(u,v)} | u = 1, \cdots, l, v = 1, \cdots, a\}$.

The attention heads and layers of the monolingual BERT [91] for the sentence *"The cat sat on the mat"* are given in Figure 4.8. Each cell in the figure shows the attention pattern for a particular layer that is indexed by the row, and the attention head that is indexed by the column. The patterns are specific to the input text.

CYK (Cocke-Younger-Kasami) [94] is used for decoding to generate the trees. The parser outputs tree $\hat{t}$ that has the lowest score:

$$\hat{t} = \arg\min_{T} s(t) \tag{11}$$

For each distance function with score functions $[JSD{-}s_p, JSD{-}s_c, HEL{-}s_p, HEL{-}s_c]$, we obtain the weights of the i^{th} layer and j^{th} attention head of that layer. Then we calculate the span scores using the distance functions. We select the tree with the lowest score for each distance function, which leads to 4 trees in i^{th} layer and j^{th} attention head. Therefore, we finally obtain $4 \times l \times a$ trees, where l is the number of layers and a is the number of attention heads. The final F1 scores are calculated for each tree and the highest F1 score is reported in the results.

4.2.3.1. Three Levels of Parsing with a Single Model We use the chart-based zero-shot parsing model for three types of parsing ranging in different semantic and syntactic levels with different granularities and structures of a given text:

- **Dependency Parsing** is concerned with the syntactic relations between words in a sentence. These syntactic relations are discovered in terms of the dependencies of words on each other. To apply the zero-shot parsing model to dependency parsing, we compute the scores for each tree and then apply Eisner [101] decoding algorithm (instead of CYK) to produce dependency trees using the tree scores.
- **Constituency Parsing** is concerned with extracting the syntactic structure of a given text through phrasal constituents. Therefore, unlike dependency parsing, it is concerned with the syntactic structure of an entire sentence rather than the relations between words as opposed to dependency parsing. We apply zero-shot parsing without adding any additional steps for constituency parsing.
- **Semantic Parsing** is concerned with extracting the semantic structure of a given text using a formal representation. In particular, we use the UCCA [17] graph-based semantic representation to extract semantic relations within the text. To perform UCCA-based parsing, we first convert UCCA graphs into constituent trees by removing discontinuities and remote edges [73, 102]. Then we perform zero-shot learning to tackle semantic parsing as a constituency parsing problem. After finding the tree with the lowest score, we convert constituency trees back to the UCCA-based

graphs and restore discontinuity units. We disregard the remote edges and implicit edges.

4.3. Experiments & Evaluation

We present the details of the datasets, the evaluation metrics, our experimental setting, and the results of our evaluation for all proposed models.

4.3.1. Datasets

We used the SemEval 2019 shared task [1] dataset and the Turkish dataset described in Chapter 5.. The details of the datasets are given in Table 4.1. The SemEval 2019 shared task [1] dataset includes the English, German, and French languages from *Wikipedia* and *Twenty Thousand Leagues Under the Sea* and the Turkish dataset includes only the Turkish language from the METU-Sabanci Turkish Treebank [103, 104].

	English-Wiki	English-20K	German-20K	French-20K	Turkish
Train	4,113	0	5,211	15	0
Validation	514	0	651	238	0
Test	515	492	652	239	400

Table 4.1 Number of sentences in each dataset in Semeval 2019 [1] and Turkish datasets

For the unsupervised parser model (see 4.2.3.), which is chart-based zero-shot parsing model, we used the following dependency and constituency datasets in addition to the semantic parsing datasets[7]:

- **Constituency Parsing:** We report the results of constituency parsing on the Penn Treebank (PTB) [105] for English, on the SPMRL dataset [106] for and German and French, and on the Turkish Annotated Treebank for Turkish [107] [8].

[7]We only use the test set of datasets for languages.

[8]`https://github.com/olcaytaner/TurkishAnnotatedTreeBank2-15`

- **Dependency Parsing:** We present the results of the model for dependency parsing on Universal Dependencies v2.3 for English, German, French and Turkish.

Here we specify the size of the test sets used in all parsing tasks.

	English	German	French	Turkish
Dependency Parsing	2077	1000	416	979
Constituency Parsing	2416	5000	2541	63
Semantic Parsing	Wiki: 515 20-K: 492	652	239	50

Table 4.2 Size of the test sets of dependency, constituency, and semantic parsing datasets used in the experiments

4.3.2. Evaluation Metrics

We followed the official evaluation metrics used in SemEval 2019 [1] for the evaluation of semantic parsing. The evaluation method measures a matching score between each output graph $G_o = (V_o, E_o, l_o)$ predicted by the model and its corresponding gold graph $G_g = (V_g, E_g, l_g)$ over the same sequence of nodes. Labeled precision and recall metrics are calculated by dividing the number of matching edges in G_o and G_g with their corresponding labels to $|E_o|$ and $|E_g|$ respectively.

F_1 is the harmonic mean of precision and recall:

$$F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision \times Recall} \tag{12}$$

Unlabeled precision, recall, and F_1 are computed analogously, but without requiring label matching for the edges. In all semantic parsing experiments, both primary and remote edges are evaluated separately.

We report the unlabeled F1 score and attachment score (UAS) for constituency parsing and dependency parsing, respectively, for the chart-based zero-shot parsing experiments.

4.3.3. Experimental Settings

We present the experimental setting of the models separately.

- **Self-Attentive Semantic Parser** We perform two single-lingual experiments, similar to Hershcovich et al. [1] for English: 1. in-domain setting using the English-Wiki corpus for both training and testing purposes (a separate validation set under the same dataset is used for testing) 2. out-of-domain setting using the English-Wiki corpus for training and the English-20K for testing purposes. We perform in-domain experiments only for English, German and French since only one dataset is available for both languages.

  The semantic parser is implemented using the PyTorch package. For the encoder, we use a self-attention layer with the same parameter values as in [92]. The word embedding dimensionality is 100 with an embedding dimensionality of 50 for PoS tags, 50 for dependency tags, 25 for entity types, and 25 for entity iob types. We used the Adam optimizer [108], cross-entropy as the objective function, and early stopping during training because of the variations in the size of the training sets.

  All syntactic embeddings (i.e., word, PoS tags, dependency tags, entity types, and entity iob types) are randomly initialized in the single-lingual experiments. In addition to syntactic embeddings, we use pre-trained fasttext [109] character n-gram based word embeddings. In addition, we use BERT [91] embeddings as contextualized embeddings to incorporate contextual information. For the cross-lingual models, we conduct experiments with and without contextual embeddings in addition to syntactic embeddings.

- **Incremental Semantic Parser** We perform the same experiments defined in the model settings of *Self-Attentive Semantic Parser*: single-lingual (in-domain and out-of-domain) and cross-lingual.

  The semantic parser is implemented using the PyTorch package. We use the Adam optimizer [108], cross-entropy as the objective function, and early stopping during training due to variations in the size of the training sets. The parameters are LSTM layer dimension 500, number of LSTM 2, word dropout 0.1, dropout 0.4, weight decay 10^{-5} and mini-batch size 32. We used pre-trained language models (BERT [91]) as word embeddings.

- **Chart-based Zero-shot Semantic Parser** In the experiments, we use both monolingual and multilingual PLMs. For English, we use the following monolingual PLMs: BERT [91], GPT-2 [110], RoBERTa [111], and XLNet [112]. We follow previous work [98–100] by using two variants of each PLM, where the X-base variant consists of 12 layers, 12 attention heads and 768 hidden dimensions, while the X-large variant has 24 layers, 16 attention heads and 1024 hidden dimensions. The GPT2 model corresponds to the X-base while GPT2-medium corresponds to the X-large model.

  We use `bert-base-german-cased`, `bert-base-french-europeana-cased`, and `bert-base-turkish-cased` for German, French, and Turkish monolingual PLMs respectively.

  For multilingual experiments, we use the multilingual version of the BERT-base model (M-BERT) [91], the XLM-base model (XLM-R) [113], which is a multilingual RoBERTa model, and the large version of XLM (XLM-R-large) [114].

4.3.4. Results

4.3.4.1. Self-Attentive Semantic Parser The results of both single-lingual and cross-lingual experiments on SemEval 2019 [1] datasets for English, French, and German are given in Table 4.3. Since the Turkish dataset is very small (400 sentences), we report results for Turkish on *Zero-shot and Few-shot Cross-Lingual Model* and *Error Analysis*.

For the single-lingual setting, the use of fasttext embeddings [115] along with BERT contextualized embeddings [91] in addition to syntactic embeddings outperforms the other settings in all languages. For the cross-lingual setting, the results have slightly decreased for

	Single-Lingual Exp.			Cross-Lingual Exp.		
	Prim.	Rem.	All	Prim.	Rem.	All
	English-Wiki					
syntactic emb.	74.5	2.1	73.04	74.8	44.7	74.19
syntactic emb. $\oplus$ fasttext	77.9	53.0	77.4	-	-	-
syntactic emb. $\oplus$ bert	78.3	52.8	77.79	79.6	48.5	78.97
syntactic emb. $\oplus$ fasttext $\oplus$ bert	80.2	55.4	**79.7**	-	-	-
	English-20K					
syntactic emb.	71.0	7.9	68.87	72.7	23.6	71.04
syntactic emb. $\oplus$ fasttext	73.8	25.0	72.15	-	-	-
syntactic emb. $\oplus$ bert	75.45	28.6	73.87	75.9	29.4	74.33
syntactic emb. $\oplus$ fasttext $\oplus$ bert	76.2	29.3	**74.62**	-	-	-
	German-20K					
syntactic emb.	77.3	31.5	76.09	80.4	49.3	79.58
syntactic emb. $\oplus$ fasttext	83.6	60.2	82.98	-	-	-
syntactic emb. $\oplus$ bert	85.1	63.7	84.54	86.2	53.6	85.34
syntactic emb. $\oplus$ fasttext $\oplus$ bert	86.7	65.1	**86.13**	-	-	-
	French-20K					
syntactic emb.	43.1	0	41.67	65.4	15.3	63.74
syntactic emb. $\oplus$ fasttext	43.2	0	41.77	-	-	-
syntactic emb. $\oplus$ bert	44.5	0	43.02	68.7	45.5	**67.93**
syntactic emb. $\oplus$ fasttext $\oplus$ bert	46.2	0	44.67	-	-	-

Table 4.3 Single-lingual and cross-lingual experimental results of self-attentive parser on Semeval 2019 dataset [1]

all languages except French. However, the results for French have improved significantly. The cross-lingual setting helps predict remote edges in French, while it is not sufficient to predict remote edges in single-lingual setting due to the insufficient amount of training data for French. We did not perform experiments with pre-trained fasttext embeddings as they are trained independently for different languages and are not available as multilingual embeddings. Using BERT improves F1 scores by about 4% for English, 5% for German, and 4% for French in the cross-lingual setting.

Comparative results of our model with other participants of Semeval 2019 [1][9] are given in Table 4.4. The results show that our model achieves state-of-the-art performance among the other parsers in English and German. The model proposed by Jiang et al. [73] outperforms

[9]We report the official results given in [1]

the other models in French. However, our results on unlabeled edges are still competitive with those of [73].

	English-Wiki					
	Labeled			Unlabeled		
	All	Prim.	Rem.	All	Prim.	Rem.
TUPA	72.8	73.3	47.2	85.0	85.8	48.4
HLT@SUDA	77.4	77.9	52.2	87.2	87.9	52.5
Davis	72.2	73.0	0	85.5	86.4	0
CUNY-PekingU	71.8	72.3	49.5	84.5	85.2	50.1
DANGNT@UIT.VNU-HCM	70.0	70.7	0	81.7	82.6	0
GCN-Sem	65.7	66.4	0	80.9	81.8	0
Self-Attentive UCCA Parser	**79.7**	80.2	55.4	**89.6**	90.3	55.3
	English-20K					
HLT@SUDA	72.7	73.6	31.2	85.2	86.4	32.1
TUPA	67.2	68.2	23.7	82.2	83.5	24.3
CUNY-PekingU	66.9	67.9	27.9	82.3	83.6	29.0
GCN-Sem	62.6	63.7	0	80.0	81.4	0
Self-Attentive UCCA Parser	**74.62**	76.2	29.3	**87.69**	89.7	30.1
	German-20K					
HLT@SUDA	84.9	85.4	64.1	92.8	93.4	64.7
TUPA	79.1	79.6	59.9	90.3	91.0	60.5
TüPa	78.1	78.8	40.8	89.4	90.3	41.2
XLangMo	78.0	78.4	61.1	89.4	90.1	61.4
MaskParse@Deskiñ	74.2	74.8	47.3	87.1	88.0	47.6
Self-Attentive UCCA Parser	**86.13**	86.7	65.1	**94.1**	94.4	64.5
	French-20K					
HLT@SUDA	**75.2**	76.0	43.3	**86.0**	87.0	45.1
XLangMo	65.6	66.6	13.3	81.5	82.8	14.1
MaskParse@Deskiñ	65.4	66.6	24.3	80.9	82.5	25.8
TUPA	48.7	49.6	2.4	74.0	75.3	3.2
TüPa	45.6	46.4	0	73.4	74.6	0
MaskParse@Deskiñ	65.4	66.6	24.3	80.9	82.5	25.8
Self-Attentive UCCA Parser	67.93	68.7	45.5	84.8	85.5	54.6

Table 4.4 Comparative F-1 results of the self-attentive parser with other participants of UCCA framework at Semeval 2019 [1]

Zero-shot and Few-shot Cross-Lingual Model: Zero-shot learning [116, 117] and few-shot learning [118] have recently shown outstanding success in various NLP tasks such as dependency parsing and text classification. Zero-shot cross-lingual model is used when no or few annotated examples are available in the target language. In contrast, the few-shot cross-lingual model is used when only a small amount of training data is available during training. Due to the insufficient size of the French and Turkish datasets, we performed both few-shot and zero-shot learning as part of the cross-lingual experiments.

In the zero-shot setting for French and Turkish, we performed cross-lingual experiments without using the dataset during training. In the few-shot setting of French, we used the train set of French during training. For Turkish, we performed 5-fold cross-validation by adding 320 sentences to the merged training set in other languages and using 80 sentences as the test set. We report the average of the scores obtained from each fold. The results are given in Table 4.5. The results show that even a small amount of data significantly improves the results in few-shot learning compared to zero-shot learning.

		Labeled			Unlabeled		
		Primary	Remote	All	Primary	Remote	All
French	single-lingual	46.2	0	44.67	68.9	0	66.62
	zero-shot	57.2	16.2	56.42	78.6	16.2	76.53
	few-shot	**68.7**	**45.5**	**67.93**	**85.8**	**54.6**	**84.48**
Turkish	zero-shot	58.2	0.0	57.9	85.9	0.0	85.7
	few-shot	**74.8**	**21.2**	**73.5**	**89.5**	**23.4**	**88.2**

Table 4.5 Effect of French and Turkish dataset on cross-lingual experiments for self-attentive parser

Error Analysis: We characterize the errors of our semantic parser by conducting further experiments to analyze the effects of structural and linguistic features of sentences on the accuracy of the parser.

- **Sentence Length:** The results for the different sentence lengths for SemEval datasets [1] are given in Table 4.6. The results show that the longer the sentences are, the lower the F-1 scores are for the remote edges except for the English-Wiki dataset. Since UCCA can be extended to represent paragraph-level annotation, the semantic structure of longer sentences can also be efficiently represented using the UCCA framework. The results obtained from the primary edges for longer sentences already confirm this. The frequency of a remote edge is 1 or 0 in each sentence in the dataset, which does not let the model learn the remote edges properly. Therefore, the efficiency of the model is more crucial for primary edges than for remote edges.

	English-Wiki			English-20K			German-20K			French-20K		
Sent. Len.	Prim.	Rem.	All	Prim.	Rem.	All	Prim.	Rem.	All	Prim.	Rem.	All
$>=10$	80.8	50.8	79.76	73.2	30.1	72.33	87.2	66.7	86.61	68.2	45.4	67.42
$>=20$	80.4	50.9	79.42	76.9	29.0	75.89	84.5	60.3	83.75	67.5	45.3	66.72
$>=30$	79.8	63.0	79.18	75.8	26.6	74.74	83.4	59.2	82.67	69.1	42.7	68.17
$>=40$	82.4	61.0	81.68	74.5	26.4	73.58	83.0	48.5	82.01	66.5	41.4	65.58
$>=50$	78.5	61.5	77.93	74.2	0	72.55	82.9	49.2	81.84	66.8	39.8	65.83

Table 4.6 Results by the length of SemEval datasets [1] on self-attentive parser

In Table 4.7, we also present the results of few-shot learning for different sentence lengths for the Turkish dataset. Since the sentences in the Turkish dataset are shorter compared to the SemEval datasets [1], the performance of the model is better for shorter sentences than for longer ones.

	Labeled			Unlabeled		
Sent. Len.	Primary	Remote	All	Primary	Remote	All
≤ 5	97.7	32.4	95.3	99.2	35.6	96.4
≤ 10	81.4	24.6	78.6	91.7	28.5	89.9
≤ 20	75.6	25.4	73.2	88.2	26.4	86.5
≤ 50	52.3	0.0	51.4	84.5	0.0	82.6

Table 4.7 Results by the length of Turkish dataset on self-attentive parser

- **Semantic Categories:** We analyze the results of each semantic category to further evaluate the performance of the model according to each category. The results of each category are given in Table 4.8. The frequency of Adverbial (D), Function (F), Ground (G), Linker (L), Connector (N), and State (S) are comparatively lower than that of the other semantic categories in the dataset[10]. While the model struggles with predicting categories with low frequency, more frequent categories are learned more accurately by the model.

	Scene Elements				Non-Scene Elements				Inter-Scene Relations			Other
dataset	P	S	A	D	C	E	N	R	H	L	G	F
English-Wiki	0.68	0.24	0.77	0.67	0.83	0.80	0.87	0.86	0.75	0.56	0.65	0.73
English-20K	0.73	0.23	0.67	0.54	0.81	0.78	0.82	0.86	0.60	0.72	0.25	0.71
German-20K	0.79	0.27	0.81	0.77	0.90	0.87	0.30	0.92	0.79	0.88	0.71	0.88
French-20K	0.68	0.24	0.58	0.32	0.78	0.71	0.75	0.83	0.49	0.59	0.46	0.41
Turkish	0.82	0.48	0.88	0.64	0.65	0.57	0	0.21	0.68	0.37	0.41	0.39

Table 4.8 F-1 measure of predicting primary edges and their labels on self-attentive parser

[10]The details of the dataset can be found in [1].

4.3.4.2. Incremental Semantic Parser The results of both single-lingual and cross-lingual experiments with the SemEval 2019 [1] datasets for English, French, and German are given in Table 4.9. For single-lingual experiments, the train set of the German dataset is larger than that of the other datasets, which is the main reason for the higher F1 score in German as defined in the literature [119]. The results of English in the in-domain setting and German have not changed in the cross-lingual setting. However, the results of English in the out-of-domain setting and French have improved significantly. Moreover, the cross-lingual setting helps predict remote edges in French, which couldn't be predicted in the single-lingual setting due to the insufficient amount of training data.

	Single-Lingual Exp.			Cross-Lingual Exp.		
	Primary	Remote	All	Primary	Remote	All
English-Wiki	78.05	52.18	76.70	77.45	53.14	**76.89**
English-20K	70.19	25.17	69.30	79.57	49.52	**78.47**
German-20K	83.15	47.18	**82.60**	83.19	48.2	82.44
French-20K	39.65	0	38.73	68.19	18.49	**67.71**

Table 4.9 Single-lingual and cross-lingual experimental results of incremental parser on Semeval 2019 dataset [1]

Comparative results of our model with other participants of Semeval 2019 [1] are given in Table 4.10. The results show that we could not achieve competitive results with the self-attentive parser and the other state-of-the-art models in the literature. Our model achieves the best results among the other parsers for English only in the out-of-domain setting. This is a very important outcome that shows the adaptation of the model for datasets from different domains.

Zero-shot and Few-shot Cross-Lingual Model: We perform both few-shot and zero-shot learning as part of the cross-lingual experiments. We use the same settings of self-attentive parser in the zero-shot and few-shot experiments. The results are given in Table 4.11. The results show that even a small amount of data significantly improves the results in few-shot learning compared to zero-shot learning similar to the results of the self-attentive parser.

	English-Wiki					
	Labeled			Unlabeled		
	All	Prim.	Rem.	All	Prim.	Rem.
TUPA	72.8	73.3	47.2	85.0	85.8	48.4
HLT@SUDA	**77.4**	**77.9**	52.2	**87.2**	**87.9**	52.5
Davis	72.2	73.0	0	85.5	86.4	0
CUNY-PekingU	71.8	72.3	49.5	84.5	85.2	50.1
DANGNT@UIT.VNU-HCM	70.0	70.7	0	81.7	82.6	0
GCN-Sem	65.7	66.4	0	80.9	81.8	0
Incremental UCCA Parser	76.89	77.45	**53.14**	82.20	83.62	**53.13**
	English-20K					
HLT@SUDA	72.7	73.6	31.2	85.2	86.4	32.1
TUPA	67.2	68.2	23.7	82.2	83.5	24.3
CUNY-PekingU	66.9	67.9	27.9	82.3	83.6	29.0
GCN-Sem	62.6	63.7	0	80.0	81.4	0
Incremental UCCA Parser	**78.47**	**79.57**	**49.52**	**89.55**	**90.72**	**50.00**
	German-20K					
HLT@SUDA	**84.9**	**85.4**	**64.1**	**92.8**	**93.4**	**64.7**
TUPA	79.1	79.6	59.9	90.3	91.0	60.5
TüPa	78.1	78.8	40.8	89.4	90.3	41.2
XLangMo	78.0	78.4	61.1	89.4	90.1	61.4
MaskParse@Deskiñ	74.2	74.8	47.3	87.1	88.0	47.6
Incremental UCCA Parser	82.60	83.15	47.18	91.29	90.60	60.67
	French-20K					
HLT@SUDA	**75.2**	**76.0**	**43.3**	**86.0**	**87.0**	**45.1**
XLangMo	65.6	66.6	13.3	81.5	82.8	14.1
MaskParse@Deskiñ	65.4	66.6	24.3	80.9	82.5	25.8
TUPA	48.7	49.6	2.4	74.0	75.3	3.2
TüPa	45.6	46.4	0	73.4	74.6	0
MaskParse@Deskiñ	65.4	66.6	24.3	80.9	82.5	25.8
Incremental UCCA Parser	67.71	68.19	18.49	80.61	82.43	32.14

Table 4.10 Comparative F-1 results of incremental parser with other participants of UCCA framework at Semeval 2019 [1]

		Labeled			Unlabeled		
		Primary	Remote	All	Primary	Remote	All
French	single-lingual	39.65	0	38.73	54.13	0	52.02
	zero-shot	67.18	13.21	66.42	81.15	30.86	80.11
	few-shot	**68.19**	**18.49**	**67.71**	**82.43**	**32.14**	**80.61**
Turkish	zero-shot	52.08	18.89	49.56	83.48	31.78	82.23
	few-shot	**75.15**	**28.75**	**74.03**	**88.07**	**48.47**	**86.92**

Table 4.11 Effect of French and Turkish dataset on cross-lingual model for incremental parser

Error Analysis: We characterize the errors of our semantic parser by conducting further experiments to analyze the effects of structural and linguistic features of sentences on the accuracy of the parser.

- **Sentence Length:** The results for the different sentence lengths for SemEval datasets [1] are given in Table 4.12. The results show that the longer the sentences are, the lower the F-1 scores are for all UCCA datasets. Unlike the self-attentive parser, which is a graph-based approach, the incremental parser does not perform better on longer sentences. The main reason for this is the propagating error in predicting transitions in long-term dependencies [120, 121].

	English-Wiki			English-20K			German-20K			French-20K		
Sent. Len.	Prim.	Rem.	All	Prim.	Rem.	All	Prim.	Rem.	All	Prim.	Rem.	All
>= 10	82.10	50.00	81.25	90.10	45.18	89.55	90.45	52.94	89.70	70.37	0	67.86
>= 20	95.15	62.15	94.60	80.80	45.83	79.84	91.41	60.00	90.95	81.13	40.00	79.90
>= 30	90.24	52.94	89.16	80.83	51.03	82.20	86.15	50.98	85.65	73.27	46.67	72.91
>= 40	91.67	50.00	90.72	77.52	51.03	78.40	88.93	39.66	87.75	75.00	18.75	73.62
>= 50	77.12	22.02	76.74	72.69	22.79	73.40	81.69	28.60	82.48	67.36	13.32	68.18

Table 4.12 Results by the length of SemEval datasets [1]

In Table 4.13, we also present the results of few-shot learning for different sentence lengths for the Turkish dataset. Since the sentences in the Turkish dataset are shorter compared to the SemEval datasets [1], the performance of the model is better for shorter sentences than for longer ones.

- **Semantic Categories:** We analyze the results of each semantic category to further evaluate the performance of the model according to each category. The results of each category are given in Table 4.14. The frequency of Adverbial (D), Function (F),

	Labeled			Unlabeled		
Sent. Len.	Primary	Remote	All	Primary	Remote	All
≤ 5	95.81	24.05	94.60	98.48	57.48	67.67
≤ 10	82.45	30.15	81.48	92.44	37.87	91.54
≤ 20	74.62	23.32	73.27	86.70	33.90	85.54
≤ 50	68.17	0	67.36	85.78	0	83.73

Table 4.13 Results by the length of Turkish dataset on incremental parser

Ground (G), Linker (L), Connector (N), and State (S) are comparatively lower than that of the other semantic categories in the dataset[11]. While the model struggles with predicting categories with low frequency, more frequent categories are learned more accurately by the model.

	Scene Elements				Non-Scene Elements				Inter-Scene Relations			Other
dataset	P	S	A	D	C	E	N	R	H	L	G	F
English-Wiki	0.60	0.22	0.75	0.94	0.84	0.81	0.84	0.83	0.74	0.55	0.62	0.70
English-20K	0.75	0.24	0.68	0.55	0.80	0.77	0.85	0.84	0.64	0.75	0.29	0.72
German-20K	0.78	0.26	0.79	0.76	0.90	0.82	0.28	0.91	0.76	0.86	0.70	0.87
French-20K	0.67	0.23	0.60	0.33	0.79	0.72	0.74	0.82	0.50	0.59	0.47	0.40
Turkish	0.83	0.52	0.87	0.63	0.66	0.59	0.	0.21	0.67	0.35	0.42	0.41

Table 4.14 F-1 measure of predicting primary edges and their labels on incremental parser

4.3.4.3. Chart-based Zero-shot Semantic Parser

We present the results obtained from each parsing separately below.

- **Dependency Parsing** Dependency Parsing results for all languages are given in Table 4.15. The best results are obtained from multilingual PLMs in all languages. Since the other unsupervised dependency parsing models are either finetuned [122, 123] or utilise other external resources such as Google Universal Treebanks [122] or WSJ [123], we have not made a comparison with other models since the model presented here is fully unsupervised, does not use any annotated data, and does not incorporate any syntactic information during PLM pre-training.

- **Constituency Parsing** For constituency parsing, we either perform top-down or chart-based parsing to generate trees. We further experiment with using different

[11]The details of the dataset can be found in [1].

Monolingual Models				
PLM	English	German	French	Turkish
BERT-base-cased[4]	26.48	26.59	24.78	35.56
BERT-large-cased	27.89	-	-	-
XLNet-base-cased	25.66	-	-	-
XLNet-large-cased	27.53	-	-	-
RoBERTa-base	27.68	-	-	-
RoBERTa-large	25.11	-	-	-
GPT2	19.66	-	-	-
GPT2-medium	21.44	-	-	-
Multilingual Models				
M-BERT	30.80	30.69	**34.37**	**41.62**
XLM-R	30.80	**31.84**	34.27	41.25
XLM-R-large	**32.66**	28.58	26.19	39.13

Table 4.15 UAS scores for dependency parsing

layers in the PLMs. All unlabeled F1 scores for the constituency parsing are given in Table 4.16 and Table 4.17. We use abbreviations TD, CP, and CC for Top-Down, Chart-Pair (pair score function $s_p(\cdot, \cdot)$) and Chart-Characteristic (characteristic score function $s_c(\cdot, \cdot)$) respectively. With the exception of English, we obtain the best results with the top-down decoder and with XLM-R for German, French, and Turkish [12].

	English			German		
PLM	TD	CP	CC	TD	CP	CC
Monolingual Models						
BERT-base-cased	34.51	40.24	42.05	26.96	24.82	26.59
BERT-large-cased	38.93	43.68	44.58	-	-	-
XLNet-base-cased	40.12	42.14	43.47	-	-	-
XLNet-large-cased	38.32	42.60	43.73	-	-	-
RoBERTa-base	40.61	45.37	**46.01**	-	-	-
RoBERTa-large	34.30	42.19	43.26	-	-	-
GPT2	34.21	34.01	35.78	-	-	-
GPT2-medium	37.65	38.59	39.81	-	-	-
Multilingual Models						
M-BERT	40.28	43.44	44.13	30.69	30.59	30.28
XLM-R	41.25	44.25	44.76	**33.13**	32.19	31.84
XLM-R-large	39.13	42.87	44.67	28.18	27.13	28.58

Table 4.16 Unlabeled F1 scores for constituency parsing in English and German

[12]The model is adopted from that of Kim et al. [99] and we prefer not to repeat the comparative scores here again.

	French			Turkish		
PLM	TD	CP	CC	TD	CP	CC
Monolingual Models						
BERT-base-cased	24.78	22.83	23.86	35.36	31.47	33.50
Multilingual Models						
M-BERT	32.88	30.37	30.45	41.29	40.61	39.93
XLM-R	**34.19**	31.29	30.93	**45.18**	43.49	42.30
XLM-R-large	26.68	25.70	26.46	36.21	36.72	36.72

Table 4.17 Unlabeled F1 scores for constituency parsing in French and Turkish

- **Semantic Parsing** The unlabeled F_1 scores for UCCA-based semantic parsing are given in Table 4.18 and 4.19. The best results are obtained from RoBERTa-base amongst the monolingual models and from XLM-R amongst the multilingual models in English. Interestingly, both RoBERTa and XLM-R give similar results. For German, French, and Turkish, the best results are obtained from multilingual models. Since this is the very first study that performs UCCA-based semantic parsing in a completely unsupervised framework, there is no other study that can be compared to ours. Therefore, we report our results only as a baseline for future studies.

The results of dependency parsing are comparatively much lower than those of constituency and semantic parsing in all languages. Unsupervised dependency parsing has mostly been performed in the literature using probabilistic generative models [124] and is comparatively harder than constituency parsing, as it requires learning finer relations between words rather than between phrases in a sentence. Interestingly, however, the results of UCCA-based semantic parsing are also promising and as good as the performance of constituency parsing. It should be noted that UCCA-based semantic parsing has not yet been tackled with an unsupervised learning model before.

As for the PLM models, the GPT and GPT2-medium perform comparatively poorly on all parsing problems. Unlike other PLMs, the GPT models are auto-regressive language models that do not allow to the incorporation of the context on both sides of a word, which might be the reason of the poor performance of the GPT models.

	English-Wiki			English-20K		
PLM	TD	CP	CC	TD	CP	CC
Monolingual Models						
BERT-base-cased	38.34	42.30	42.60	39.10	42.80	43.93
BERT-large-cased	38.33	42.93	43.52	39.41	43.82	44.75
XLNet-base-cased	37.00	39.62	40.18	37.57	39.56	42.70
XLNet-large-cased	38.41	41.25	42.27	39.98	41.20	41.52
RoBERTa-base	41.82	44.96	**45.21**	32.43	45.62	**46.18**
RoBERTa-large	37.65	41.37	41.62	36.44	41.78	41.92
GPT2	31.97	38.23	38.56	32.41	37.97	38.40
GPT2-medium	34.86	38.49	38.58	32.21	37.68	39.31
Multilingual Models						
M-BERT	39.62	43.52	44.06	38.11	43.99	45.15
XLM-R	40.98	45.45	**45.89**	42.06	45.51	**46.30**
XLM-R-large	36.40	40.05	40.87	33.69	40.00	41.45

Table 4.18 Unlabeled F1 scores for semantic parsing in English (English-Wiki, English-20K)

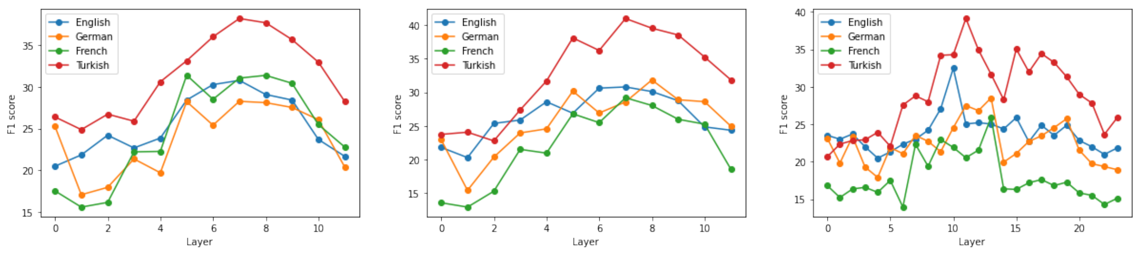
	German-20K			French-20K			Turkish		
PLM	TD	CP	CC	TD	CP	CC	TD	CP	CC
Monolingual Models									
BERT-base-cased[13]	40.30	41.93	42.96	40.32	40.55	42.71	38.78	41.10	40.33
Multilingual Models									
M-BERT	39.08	**44.17**	44.07	41.01	43.26	46.08	41.18	38.86	39.32
XLM-R	40.90	43.15	42.98	44.13	46.08	**47.38**	**47.38**	46.37	46.37
XLM-R-large	35.59	39.63	42.37	37.56	39.17	38.94	44.96	40.64	40.64

Table 4.19 Unlabeled F1 scores for semantic parsing in German, French and Turkish

Analysis of the Results

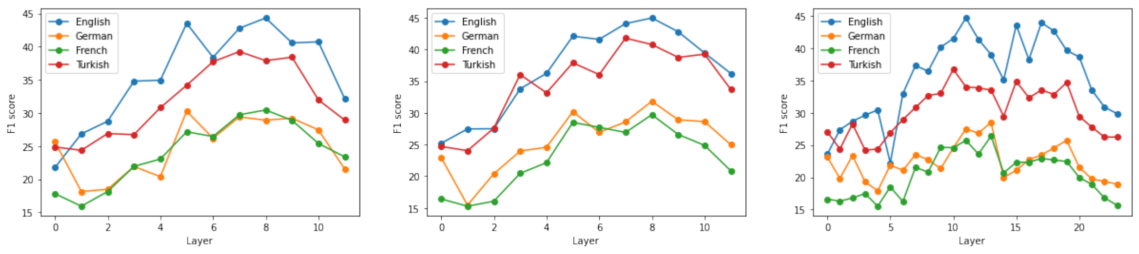
We analyse the attention layers and heads that contribute most to each parsing task, along with the effect of sentence length in the experiments.

- **Attention Layers** We analyse the attention layers to see which layers provide the most information for the parsing tasks.
  - **Dependency Parsing** UAS scores obtained from multilingual models for each layer are illustrated in Figure 4.9. The results show that we get the highest UAS scores from the middle or the ones closer to the final layers of the PLMs for all languages.



(a) Bert-multilingual (b) XLM-R (c) XLM-R-large

Figure 4.9 UAS scores of multilingual PLMs for dependency parsing



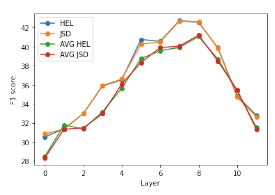
(a) Bert-multilingual (b) XLM-R (c) XLM-R-large

Figure 4.10 F1 scores of multilingual PLMs for constituency parsing

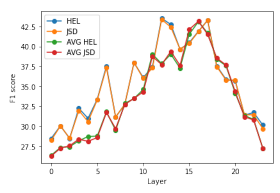
- **Constituency Parsing** F1 scores obtained from multilingual PLMs for all layers are given in Figure 4.10. Although there are slight differences between languages, the general picture does not differ from the dependency parsing results, and again the highest scores are mainly obtained from the middle layers.

- **Semantic Parsing** F1 scores obtained from monolingual models for all layers along with the different distance functions are given in Figure 4.11. Only the best scores obtained from the attentions in each layer are illustrated. The graphs show that there is not much difference between the distance functions in terms of their performance in parsing. However, we obtain the highest scores again from the middle or towards the last layers, except for GPT-2, which achieves the best results in the lower layers.

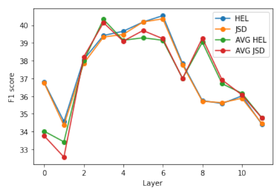
• **Attention Heads** We also analyse the attention heads in the layers to observe which attention heads contribute the most to each parsing task. F1 scores obtained from the attention heads in the different layers are given in Figure 4.12, Figure 4.13,



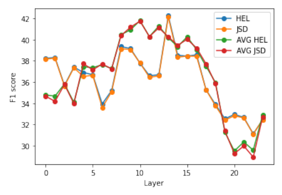
(a) Bert-base



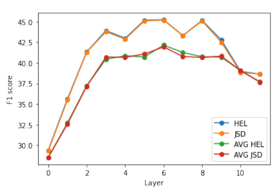
(b) Bert-large



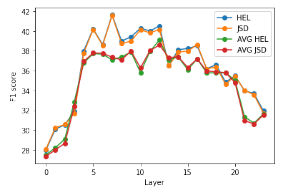
(c) XLNet-base



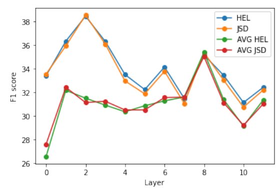
(d) XLNet-large



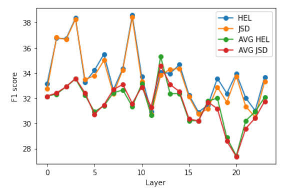
(e) RoBERTa-base



(f) RoBERTa-large

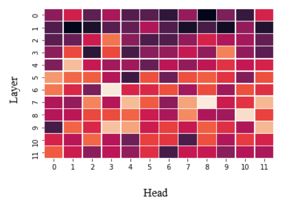


(g) GPT2

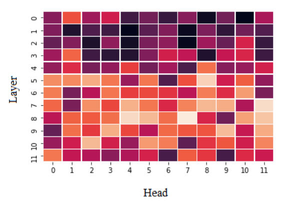


(h) GPT2-medium

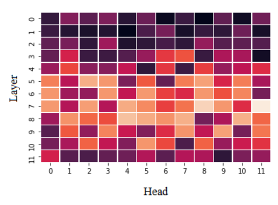
Figure 4.11 F1 scores from monolingual PLMs using the English-Wiki dataset for semantic parsing



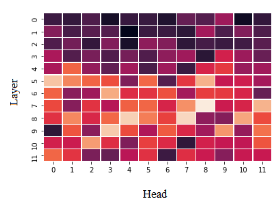
(a) English



(b) German



(c) French

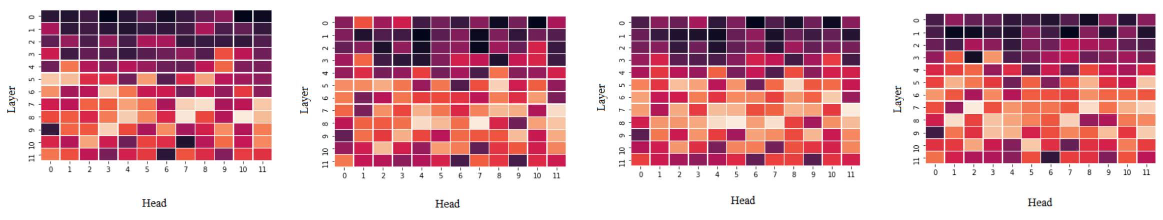


(d) Turkish

Figure 4.12 Unsupervised dependency parsing performance in all languages according to different attention heads and hidden layers with HEL distance function (Light cells refer to higher UAS scores)

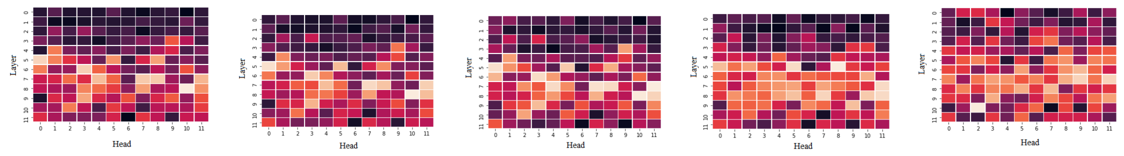
and Figure 4.14 for dependency, constituency, and semantic parsing (with XLM-R) respectively. The graphs support the findings regarding the hidden layers and also show that the top heads contain more information for all tasks and languages, except for Turkish constituency and semantic parsing, where the lower heads contain more information. This could again be due to the length of the sentences in the Turkish datasets.

- **Multilinguality** The results of the multilingual PLMs for all languages are given in Figure 4.15. The F1 scores of the languages are very low in the first hidden layers except for Turkish. The lower hidden layers could be more informative for short sentences since the Turkish UCCA dataset involves shorter sentences compared to other languages. This could be the reason for this difference between languages. The



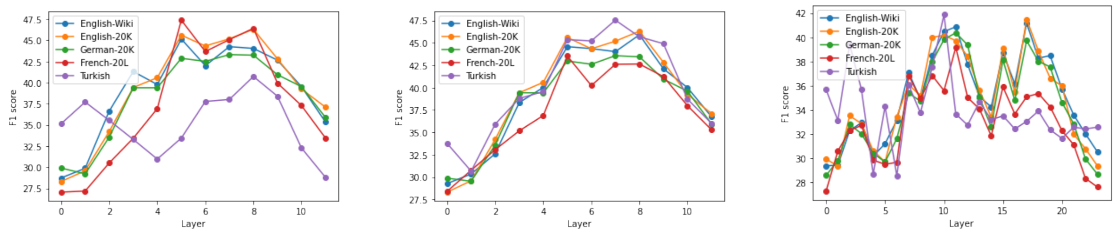
(a) English (b) German (c) French (d) Turkish

Figure 4.13 Unsupervised constituency parsing performance in all languages for different attention heads and hidden layers with HEL distance function (Light cells correspond to higher F1 scores)



(a) English-Wiki (b) English-20K (c) German-20K (d) French-20K (e) Turkish

Figure 4.14 Unsupervised UCCA semantic parsing performance in all languages with different attention heads and hidden layers with HEL distance function (Light cells refer to higher F1 scores)

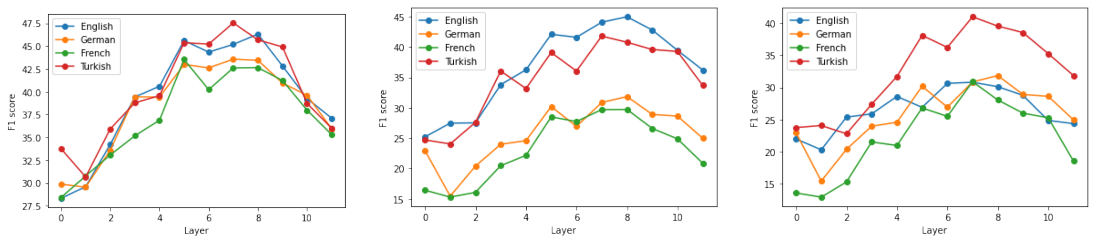


(a) Bert-multilingual (b) XLM-R (c) XLM-R-large

Figure 4.15 F1 scores from multilingual PLMs using the UCCA datasets

results also support that the final layers bear more syntactic information compared to the lower layers, especially for longer sentences, which is consistent with the findings of other studies [98, 99, 125].

To analyze and compare the PLMs for the parsing problems, we look in particular at the XLM-R multilingual PLM, since it generally outperforms the other models. The comparative results are given in Figure 4.16. It can be clearly seen that the PLMs for Turkish are informative for all parsing problems. Moreover, the results of the parsing problems for French and German are all similar.



(a) Semantic Parsing (b) Constituency Parsing (c) Dependency Parsing

Figure 4.16 F1 scores of XML-R for all problems

		samples < average		samples > average	
Language	Av. Len	Av. Len	F1	Av. Len	F1
English	20.46	12.76	43.73	28.96	39.04
German	18.82	11.62	33.71	30.07	30.37
French	29.73	16.82	34.64	46.05	31.85
Turkish	13.95	11.84	48.42	16.48	44.29

Table 4.20 F1 scores of constituency parsing for samples that are shorter and longer than the average overall sentence length

- **Sentence Length** To understand the effect of sentence length, we extract the average length of sentences in all datasets. The average sentence length of Turkish datasets for all tasks is less than that of the other languages, whereas the average sentence length of German and French is higher in all parsing datasets.

  To investigate the relationship between sentence length and the accuracy of the parsing, we run constituency parsing with XLM-R multilingual PLM and top-down parser on 1000 samples with length less than the average length of the dataset and 1000 samples with length greater than the average length of the datasets in English, French, and German. We only use 50 samples (25 less and 25 are greater than the average length in Turkish since there are only 63 samples in the dataset. Table 4.20 gives the average length of the sentences in each dataset along with the obtained F1 scores. The results show that the model performs better on shorter sentences. This also confirms that the model can hardly find distant relationships in longer sentences.

4.4. Discussion

The answer to the questions we asked at the beginning of this chapter.

1. Which approach (transition-based, graph-based) is most effective for the UCCA-based semantic parsing problem?

   To answer this question, we propose self-attentive and incremental parsers as graph-based and transition-based parsers, respectively. The self-attentive parser is based on an encoder-decoder architecture, where the encoder of the Transformer [92] with 2 MLP classifiers with 2 fully-encoded layers and ReLU activation function is the encoder and CYK is the decoder. The transition-based model is based on the BiLSTM layer and the adaptation of the `attach-juxtapose` transition set for the UCCA representation. The experimental results show that the graph-based semantic parser outperforms the transition-based approach in all languages except English with out-of-domain setting. This conclusion has also been proven in previous studies using graph-based approaches [71, 73, 80]. For longer sentences, the self-attentive parser performs better than the incremental parser compared to shorter sentences. This is evidence that transition-based parsers need a solution for longer sentences, since predicting long-term dependencies is a challenge for transition-based parsers.

2. Are the constituency parsing models applicable to the semantic parsing problems after preprocessing the UCCA representation?

   We adopt self-attentive constituency parser proposed by Kitaev and Klein [93] for semantic parsing. Since the UCCA representation is in the DAG formalism due to the remote edges, we apply a preprocessing step to convert the UCCA representations into constituency trees. We remove remote edges and add an extra MLP layer for predicting remote edges and push the labels of the edges to the nodes as in the constituency tree. In this way, we can easily apply the constituency parsing model for UCCA parsing. There are also studies in the literature that use constituency parsing models for UCCA parsing [73, 80] and the results of these models are comparatively high for the problem.

This is the main motivation for our self-attentive UCCA parser in this thesis. We obtain the best results for the English and German datasets of the SemEval 2019 [10] dataset. These datasets are comparatively larger than the French dataset. We conclude that the self-attentive encoder requires more train set to capture more information for the words in the datasets.

3. Is the transition set of the transition-based parser important to learn the features of the UCCA representation (discontinuous units, reentrancy, etc.)?

   We adopt the `attach-juxtapose` system as a transition-based UCCA parser using the same architecture of the TUPA parser [126], the first proposed transition-based UCCA parser, and compare the results. The incremental parser outperforms the TUPA parser [126] and demonstrates the importance of the transition set in a transition-based parser. In particular, the incremental parser achieves the highest score in German and competes with the graph-based parser. The model also outperforms the graph-based parser in the English dataset in the out-of-domain setting. This shows that a transition-based parser with an efficient transition set is adaptable for out-of-domain datasets.

4. Given the problem of data scarcity in UCCA representation, up to what extent does an unsupervised model learn?

   Unsupervised models are very important in NLP tasks with the problem of lack of data. The UCCA semantic representation is a comparatively new representation compared to others (AMR, UDS, etc.) and the datasets are released for limited languages (English, German, French, etc.). There are supervised studies on semantic parsing for UCCA [46, 73, 80]. However, UCCA parsing has never been studied in an unsupervised manner. There are studies in the literature that use PLMs for NLP problems [98, 100] since PLMs contain enough syntactic knowledge in different layers and heads for these problems. Therefore, we adopted chart-based zero-shot parsing [98–100] for semantic parsing by comparing constituency and dependency parsing. The results are lower than those of the supervised models, but we get an insight into the information of the PLMS weights for semantic parsing. For semantic parsing, we obtain similar results for all languages with different sentence lengths. Moreover, the best results are obtained in multilingual models.

4.5. Summary

In this chapter, we focus on the problem of the semantic parser from different perspectives. We introduced models for UCCA parsing to investigate the difference between transition and graph-based approaches, and supervised and unsupervised DL models. We showed that the graph-based approach named self-attentive parser, which solves UCCA parsing as constituency tree parser using self-attentive encoder and the CYK decoder, outperforms the transition-based parser called incremental parser using a BiLSTM encoder with an extended transition set. These results support the studies [73, 80] in the literature with higher results for all languages.

We investigated the unsupervised parser model, called the chart-based zero-shot parser, using the syntactic knowledge of PLMs. Using different distance functions, we tested the model on different parsing problems, constituency, dependency, and semantic parsers, at different semantic and syntactic levels. The results of the unsupervised model are obviously lower than those of the supervised models. However, there are insights into the PLMs for the problem that can be improved in a semi-supervised setting.

Our contributions in this chapter are as follows:

- We proposed 3 neural semantic parser models that learn UCCA-based semantic representations of sentences, 2 of them supervised and one of them unsupervised.
- We conducted single-lingual and cross-lingual experiments for the semantic parsing task with the supervised semantic parser models. For the cross-lingual experiments, we performed both few-shot and zero-shot learning due to the insufficient size of the available training data in French and Turkish.

- We obtained state-of-the-art semantic parsing results in English and German in single-lingual setting with the self-attentive semantic parser, and in English out-of-domain experiments in cross-lingual setting with the incremental parser. The results showed that the cross-lingual model performs better in low-resource languages.
- We introduced the unsupervised semantic parser model, which is the first attempt for UCCA representation in an unsupervised setting. The model used distance functions to compute scores from syntactic information learned by transformer-based PLMs. We used the same model with different decoders specialized for various parsing problems, namely dependency, constituency, and semantic parsing. To the best of our knowledge, this is the first study that compares an unsupervised model for three different parsing problems in a fully unsupervised setting and analyses the linguistic information learned from PLMs during pre-training for three different parsing tasks from syntax to semantics.

5. ANNOTATION OF A TURKISH UCCA DATASET

UCCA is a recently proposed semantic annotation framework for representing the "semantic meaning" of a sentence within a multi-layered framework, where each layer corresponds to a semantic module. The ultimate goal of UCCA is to provide a semantic representation, applicable across languages, that enables parsing across languages using cross-lingual machine learning (ML) approaches, as well as parsing across different domains. It also supports rapid annotation by non-experts who do not have a proficient linguistic background. Due to these advantages, the UCCA representation has gained remarkable attention and has been part of the recent shared tasks such as SemEval 2019 [1], MRP 2019 [40] and MRP 2020 [64].

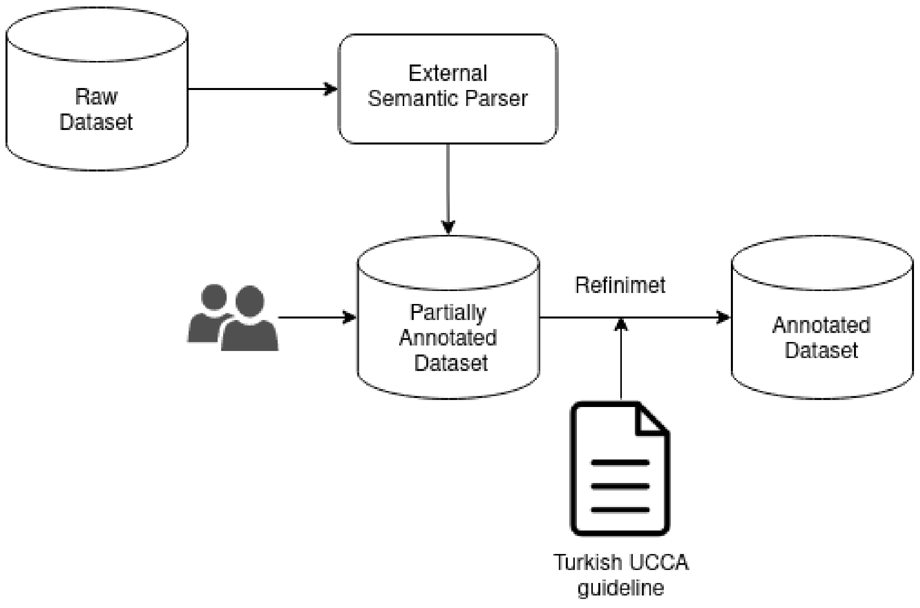


Figure 5.1 Turkish UCCA dataset annotation procedure

In this chapter, we aim to annotate a Turkish UCCA dataset in a semi-automatic pipeline by applying the self-attentive semantic parser proposed in Chapter 4.2.1. in zero-shot learning. The annotation procedure consists of two steps: (1) An external semantic parser is trained on a dataset that is a combination of English, German and French UCCA annotated datasets released in SemEval 2019 [1] to produce semantic representations that are partially correct, since there is no annotated dataset in Turkish. (2) The semantic representations obtained from the semantic parser are manually corrected as needed, which also helps to define UCCA annotation rules in line with Turkish syntax. The annotation pipeline is illustrated in Figure 5.1.

5.1. Related Work

5.1.1. Turkish Datasets

Turkish is an agglutinative language in which words are formed by productive affixation and multiple suffixes can be attached to a root to form a new word form (e.g., gidebilirsek, which means “if we can go” in English). It is possible to generate an infinite number of words in Turkish, as shown by Sak et al. [127]. This poses a challenge in many NLP tasks, such as language modeling, spell checking, and NMT, because of the out-of-vocabulary (OOV) problem. Turkish grammar presents also other challenges, such as the free word order in a sentence and the use of clitics [128, 129] that may have various meanings depending on the context.

The datasets for many NLP tasks have been released mostly in English. The datasets that have been released in Turkish are very limited. Current Turkish NLP studies generally focus on building datasets for syntactic parsing, such as the METU-Sabanci Treebank [103, 104] and the IMST Dependency Treebank [130]. Although semantic annotations are crucial for NLP tasks, there are few studies on Turkish semantic annotation [32, 129, 131]. One of the semantic datasets in Turkish is the Turkish Proposition Bank (PropBank) [131], the first semantically annotated corpus built particularly for semantic role labeling (SRL). The other annotated dataset in Turkish is the AMR corpus presented by Azin and Eryiğit [129] and Oral et al. [32]. Azin and Eryiğit [129] presented the preliminary investigation on Turkish AMR with 100 annotated sentences obtained from the Turkish translation of the novel *Little Prince* based on AMR specifications to demonstrate the differences between the Turkish and English annotations. Oral et al. [32] extended the annotation process by converting the annotation process into a semi-automatic annotation using a rule-based parser in which ProbBank [131] sentences are converted into AMR graphs. Human annotators used the output of the rule-based parser to build the dataset rather than annotating them from the very beginning. The presented AMR corpus contains 700 sentences (100 sentences from the Turkish translation of the novel *Little Prince* and 600 sentences from the IMST Dependency Treebank [130–132]).

5.1.2. UCCA Annotated Datasets

Since UCCA was first proposed by Abend and Rappoport [17], English UCCA annotated datasets, which are *English Wikipedia* [39] and *English 20K Leagues Under The Sea*, were released [133] along with an annotation guideline[14]. These datasets were followed by other UCCA annotated datasets in several languages, including French, German, Russian, and Hebrew. For English, the datasets are obtained from *Wikipedia* [39], *Web Treebank* [134], *The Little Prince* [64] and an English-French parallel corpus of *Twenty Thousand Leagues Under the Sea* [133] including the first five chapters. The expansion of the UCCA dataset to other languages started with the parallel corpus of *Twenty Thousand Leagues Under the Sea*. The German dataset [1] consists of the entire book, while the French dataset contains the first five chapters of the parallel corpus annotated using cross-lingual methods [133]. In addition, the book *The Little Prince* is used for German [64], Russian [1], and Hebrew, the last two languages being new to UCCA datasets.

5.2. Turkish Annotation Guideline

After obtaining the partially correct annotations from the external semantic parser trained in zero-shot learning (see details in Chapter 4.2.1.), we manually analysed the output of the parser and defined new rules in addition to the existing rules in the English UCCA guideline [135], especially for the cases where the rules do not cover Turkish grammar. We do not describe the existing UCCA annotation rules again here, but only describe the new UCCA annotation rules that were encountered during annotation.

[14]`https://universalconceptualcognitiveannotation.github.io/`

- **Complex Sentences:** Sentences in Turkish, as in other languages, can be either simple or complex. In English, clauses are defined as E-Scene, A-Scene or Parallel Scene in the UCCA annotation. However, there are also clauses called Adverbial clauses that define verbs in Turkish. Therefore, we have added D-Scene to label these clauses in the sentences. You can find examples of all types of clauses in Turkish, including Adverbial clauses[15][16].

  Examples:

  – $\langle$Seni üzmekten$\rangle_A$ korkuyorum (in English, *"I am afraid $\langle$to upset you$\rangle$"*)

  – $\langle$Benden sonra aşık olduğu$\rangle_E$ adamı gece gündüz izledim (in English, *"I watched $\langle$the man she had fallen in love with after me$\rangle$ day and night"*)

  – $\langle$Dönüp$\rangle_H$ $\langle$baktı bana$\rangle_H$ (in English, *"$\langle$S/he turned$\rangle$ and $\langle$looked at me$\rangle$"*)

  – Dar yollarda $\langle$koşarak$\rangle_D$ giden Kerem'i yakaladım (in English, *"I caught Kerem $\langle$running$\rangle$ on narrow roads"*)

- **Pronoun-dropping [136]:** Pronoun subjects may be omitted in a sentence since Turkish is a pro-drop language. Omitted pronoun subjects are marked with the label A-IMPLICIT.

  Examples:

  – $\langle$(Ben)[17]$\rangle_{A-IMPLICIT}$ bilmiyorum (in English, *"$\langle$I$\rangle$ don't know"*)

  – $\langle$(Biz)$\rangle_{A-IMPLICIT}$ bir süre sessiz yürüdük (in English, *"$\langle$We$\rangle$ walked in silence for a while"*)

- **Genitive case:** To express possession in Turkish, the genitive case suffix is attached to the possessor and the possessive suffix is attached to the possessed noun. Pronominal possessors of possessive nouns can also be omitted since the possessive suffix already

[15]The word or groups of words defining the semantic label and the corresponding words in English are indicated in the examples by $\langle\rangle$.
[16]Please note that no punctuation is given in the example sentences for simplicity.
[17]The word between () indicates the omitted word in the examples.

contains the possessive meaning. Omitted pronominal possessors of possessive nouns are labeled with E-IMPLICIT.

Examples:

- $\langle$(Onun)$\rangle_{E-IMPLICIT}$ gözleri buğulanmıştı bir an (in English, *"$\langle$Her/his$\rangle$ eyes were fogged for a moment"*)
- $\langle$(Onun)$\rangle_{E-IMPLICIT}$ kumral saçları hafifçe karışmıştı (in English, *"$\langle$Her/his$\rangle$ brown hair was slightly tangled"*)

- **Inflectional suffixes:** While inflectional suffixes are added to the end of the word when it is a common noun (e.g., "kedi" (cat)), they are separated from the proper noun (e.g., "Kerem", "İstanbul") by an apostrophe. Since they relate the word to the Scene, we separated inflectional suffixes from the proper nouns and labeled them as Relator. However, we left the common nouns as they are.

  Examples:

  - Erkekler Parkı '$\langle$na$\rangle_R$ gidiyorsun (in English, *"Your are going $\langle$to$\rangle$ the Erkekler Park"*)
  - Milas '$\langle$lı$\rangle_R$ (in English, *"The one $\langle$from$\rangle$ Milas"*)

- **Negation** In Turkish, verbal negation is imposed by adding a suffix to the end of the verb. However, to negate a nominal sentence, the word "değil" (not) is used within the sentence. The negation word is marked as Adverbial, as defined in the English guideline.

  Example:

  - Ne tuhaf şey $\langle$değil$\rangle_D$ mi (in English, *"What a strange thing is $\langle$not$\rangle$ it"*)

- **Reduplication:** In Turkish, there are three types of reduplication, which are defined below:

  - **Emphatic Reduplication:** It is used to emphasise the quality of an adjective ("dar" (narrow) → "da**p**dar" (very narrow)). We label the word with its actual category as defined in the UCCA guideline, since it does not require any additional rule[18].

Example:

  * $\langle$Simsiyah$\rangle_D$ olmaya başladı (in English, *"It started to turn $\langle$pitch black$\rangle$"*)

- **m-Reduplication:** In this form, a word is duplicated by adding 'm' letter to the beginning of the second word, either by replacing the first consonant with 'm' or by adding 'm' when the word begins with a vowel. We combine the reduplicated form with the word during annotation, since the reduplicated form is usually not a valid word (e.g., "etek metek" (skirt and the like))[18].

  Example:

  * $\langle$Etek metek$\rangle_A$ almadı (in English, *"S/he didn't buy $\langle$skirt and the like$\rangle$"*)

- **Doubling:** In this form, a word is doubled with the same form of the word (e.g., "koşa koşa" (in a hurry or willingly, depending on the context)). Analogously, we combine the doubled form with the word to be concise with the annotation. We label the word with its actual category as also defined in the UCCA guideline.

  Examples:

  * $\langle$Koşa koşa$\rangle_D$ geldim buraya (in English, *"I came here $\langle$in a hurry$\rangle$"*)
  * $\langle$Hızlı hızlı$\rangle_D$ yürümeye başlamıştı (in English, *"S/he was walking $\langle$fast$\rangle$"*)

• **Auxiliary verbs:** In Turkish, auxiliary verbs are usually attached to the main verb or nominal, called bound auxiliary verbs. There are also free auxiliary verbs such as "ol-" (to be), "et-" (to do), "gel-" (to come), "dur-" (to stop), "kal-" (to stay), "düş-" (to fall) etc. Bound auxiliary verbs are attached to the verb in the form of suffixes. We decided not to label them separately because morpheme-level annotation is out of scope in this study. We only label free auxiliary verbs as `Adverbial`, similar to the actual guideline.

[18]During annotation, we did not come across any example of this type of marker. Therefore, the example does not exist in the METU dataset.

Examples:

- Özgür $\langle$kalmak$\rangle_D$ istemiyorum (in English, *"I don't want $\langle$to be$\rangle$ free"*)
- Orada $\langle$olmalıyım$\rangle_D$ şimdi (in English, *"I $\langle$should be$\rangle$ there now"*)

• **Postpositions:** Unlike English, Turkish does not have prepositions [57]. Some English prepositions correspond to inflectional suffixes in Turkish (e.g., "okul**da**" (**at** the school), "arkadaşı**yla**" (**with** her/his friend)). The other prepositions are seen in the form of postpositions that follow their complement phrases. Such postpositions are identified with the label `Relator` because they relate the complement phrases to the Scene.

Examples:

- *Gözleri kor $\langle$gibi$\rangle_R$ yanıyordu* (in English, *"Her/his eyes were burning $\langle$like$\rangle$ embers"*)
- Benden $\langle$sonra$\rangle_R$ aşık olduğu adamı izledim gece gündüz (in English, *"I watched the man she fell in love with $\langle$after$\rangle$ me day and night"*)
- Ruhunun $\langle$içine$\rangle_R$ girdik (in English, *"We got $\langle$inside$\rangle$ her/his soul"*)
- Gözlerinde korku ve acı $\langle$ile$\rangle_R$ bize bakıyordu (in English, *"S/he was looking at us $\langle$with$\rangle$ fear and pain in her eyes"*)
- Arkadaşı $\langle$için$\rangle_R$ buraya geldi[18] (in English, *"She came here $\langle$for$\rangle$ her friend"*)

• **Juxtaposition:** One of the most common methods to co-ordinate two or more phrases or sentences is simply to list them without using a connector. We combined such phrases and labeled them as a whole. We also labeled each of them as a `Parallel Scene`.

Examples:

- $\langle$Bağışlayın$\rangle_H$ $\langle$koşa koşa geldim buraya$\rangle_H$ (in English, *"$\langle$Forgive me$\rangle$ $\langle$I came running here$\rangle$"*)

  - $\langle$Onu elinden kaçırmış$\rangle$ $\langle$onu bir başka erkeğe kaptırmıştı$\rangle_H$ (in English, *“$\langle$S/he missed her/him$\rangle$ $\langle$S/he had lost her/him to another man$\rangle$”*)

- **Clitics** Clitics are usually in the form of a free morpheme (e.g., *mI*, *dA*, *ya*, *ki*, and *bile*, *ile*). They have to be carefully annotated, as they may have different meanings in different contexts.

  - “**mI**”[19] can be used in 3 different meanings.

    1. **Yes/No condition:** It is used to make a question sentence. In this case, it is labeled with a `Function` since it does not refer to a participant or a relation.
       Example:
       * Sakinleştin $\langle$mi$\rangle_F$ biraz (in English, *“$\langle$Have$\rangle$ you calmed down a bit?”*)
    2. **Adverbial clause marker:**It has the meaning of *as soon as* or *once*, and connects two clauses. Therefore, it is labeled as `Linker` which corresponds to the same category in the UCCA guideline[18].
       Example:
       * Okula gittin $\langle$mi$\rangle_L$ arkadaş edinirsin (in English, *“$\langle$When$\rangle$ you go to school, you make friends”*)
    3. **Intensifier in doubled forms:** It is used in a doubled form to connect the two same adjectives in order to intensify the quantity. We combine the doubled forms (e.g., “karanlık mı karanlık” (in English, very dark)) and label as a whole[18].
       Example:
       * $\langle$Karanlık mı karanlık$\rangle_E$ bir yolda yürüyoruz (in English, *“We are walking on a $\langle$very dark$\rangle$ road”*)

  - “**dA**” can be used in 6 different meanings.

    1. **Additive Function:** It is labeled as an `Adverbial` since it attributes the meaning of “moreover” to a sentence or a clause[18].

[19]The clitic also involves the other forms of “mı”, such as “mi”, “mısın”, “mısınız” etc depending on the vowel harmony and the person type.

Example:

- $*$ Filmi izlemedim $\langle$de$\rangle_D$ (in English, *"$\langle$Morover$\rangle$ I did not watch the movie"*)

2. **Adversative Function:** It is labeled as a `Linker` since it links two or more sentences with the meaning of "but" [18].

   Example:

   - $*$ Filmi izlemedim $\langle$de$\rangle_L$ anlattılar (in English, *"I didn't watch the movie $\langle$but$\rangle$ they told me"* [20])

3. **Continuative/topic shifting:** It has the meaning of "also" or "either", depending on the position of the clitic in the sentence. It is labeled as `Elaborator` if the clitic refers to the Participant, and as `Adverbial` if it refers to the adverb.

   Examples:

   - $*$ O $\langle$da$\rangle_E$ burada bekler (in English, *"S/he $\langle$also$\rangle$ waits here"*)
   - $*$ Sanırım o zaman $\langle$da$\rangle_D$ gelmemişti (in English, *"I guess s/he didn't come then $\langle$either$\rangle$"*)

4. **Enumerating:** This role is similar to Continuative/topic shifting and it is also labeled as `Elaborator`[18].

   Example:

   - $*$ Ayşe $\langle$de$\rangle_E$ Sema $\langle$da$\rangle_E$ filmi izledier (in English, *"$\langle$Both$\rangle$ Ayşe $\langle$and$\rangle$ Sema watched the movie"*)

5. **Modifier of Adverb** It is labeled as `Adverbial` since it modifies the Adverbial in the annotation.

   Example:

   - $*$ Şimdi adımlarını daha $\langle$da$\rangle_D$ hızlandırmıştı (in English, *"Now s/he had accelerated her steps $\langle$even$\rangle$ more"*)

[20]The word "me" does not correspond to a word in Turkish, but it is expressed rather implicitly as a morpheme in the verb "anlattılar" (in English, *"they told me"*)

6. **Discourse Connective** It is labeled as `Linker` since it connects Scenes in the sentence.

   Example:

   * Şimdi düşünüyorum $\langle$da$\rangle_L$ galiba o parkın dışında yapamayacağım ben dedi Kerem (in English, *"Now I'm thinking about it, $\langle$and$\rangle$ I guess I can't do it outside of that park, Kerem said"*[21])

– "**ya**" has 4 different meanings.

1. **Contrastive adversative conjunction:** It is labeled as a `Linker` that adds a contrastive meaning such as "but"[18].

   Example:

   * İzledim $\langle$ya$\rangle_L$ anlamadım (in English, *"I watched $\langle$but$\rangle$ couldn't understand it"*)

2. **Repudiative discourse connective:** It generally occurs at the end of a sentence which is usually punctuated with an exclamation mark to express the speaker's opinion with a firm tone. Therefore, it is labeled as `Ground`[18].

   Example:

   * Gitmedim dedim $\langle$ya$\rangle_G$ (in English *"I told you I did not go"*)

3. **Reminding discourse connective:** It generally occurs in a Scene-final position that has the same role with the Repudiative discourse connective but this time it is used for reminding purposes. Therefore, it is also labeled as `Ground`[18].

   Example:

   * [Sana söylemiştim $\langle$ya$\rangle_G$]$_H$ [işte o okul]$_H$ (in English, *"I told you that is the school"*)

4. **Stressable discourse connective:** It precedes a phrase and introduces an alternative question. Since it precedes a phrase, we label it as a `Linker`[18].

[21]The word "and" does not correspond to a word in Turkish, but it is expressed by clitic "da".

Example:

- $*$ $\langle$Ya$\rangle_L$ orayı bir daha bulamazsam (in English, *“$\langle$What if$\rangle$ I cannot find it again”*)

– **ki** has 4 different meanings.

1. **Subordinator connective:** It connects a noun, an adverbial clause, or a clause with a sentence. If it connects a noun or a noun clause it is labeled as a `Relator`, otherwise as a `Linker`.

   Examples:

   - $*$ İnanıyorum $\langle$ki$\rangle_R$ onlar gelecek (in English, *“I believe $\langle$that$\rangle$ they will come”*)
   - $*$ O kadar iyisin $\langle$ki$\rangle_L$ özgürlüğüm kısıtlanıyor istediğim gibi davranamıyorum (in English, *“You’re so good $\langle$that$\rangle$ my freedom is being restricted I can’t act as I want”*)

2. **Repudiative discourse connective:** Since it attributes the meaning of “just” or “such” to a sentence and expresses the opinion of the speaker, it is labeled as `Ground`.

   Example:

   - $*$ Öyle bir şey $\langle$ki$\rangle_G$ (in English, *“It is such a thing”*)

3. **Exclamations:** It is labeled as a `Ground` since it has the meanings of “o kadar” (such) or “öyle(sine)” (so), and expresses the speaker’s opinion.

   Example:

   - $*$ Anlatacağım öyle bir şey $\langle$ki$\rangle_G$ (in English, *“I will tell something like that”*)

4. **Relative clause marker**: It has two types: non-restrictive relative clauses and restrictive relative clauses. Both types connect a clause with a sentence. Therefore, it is labeled as a `Relator`.

   Examples:

   - $*$ Ahmet $\langle$ki$\rangle_R$ müzik sevmez o bile geldi (in English *“Even Ahmet $\langle$who$\rangle$ does not like music came”*)

- $*$ Bugün okula gelirse $\langle$ki$\rangle_R$ geleceğini hiç sanmıyorum oyuna gidecekmiş (in English *"If s/he will come to the school today $\langle$which$\rangle$ I don't think s/he will s/he will go to the game"*)

– "**bile**" It is an additive connective that has the meanings such as "already" or "even". Therefore, it is labeled as an `Adverbial`[18].

  Example:

  - $*$ Gönderdim $\langle$bile$\rangle_D$ (in English *"I have $\langle$already$\rangle$ sent it"*)

- **Subordinators:** Subordinators link the clauses to superordinate clauses. The subordinators in Turkish are "diye", "ki", "mI" and "dA", as well as other obsolescent subordinators that contain "ki' ("ola ki", "meğer ki" etc.). Since we already described the other clitics above, we describe only "diye" here.

  1. "**diye**" It relates the clause to a superordinate clause. Therefore, it is labeled as `Relator`.

     Example:

     – Nereden biliyorsunuz $\langle$diye$\rangle_R$ sordu (in English *"He asked $\langle$(that)$\rangle_R$[22] how do you know"*)

- **Nominal sentences:** In Turkish, nominal sentences do not contain an explicit verb. If the predicate is not indicated by copular markers (e.g., "-(y)DI" (was), "-(y)mIş" (was), and "-(y)sA" (if it is)), we inserted a generalizing modality marker '-DIr" (is) as an implicit unit.

  Examples:

  – Nasıl bir kadın $\langle$(dır)$\rangle_{S-IMPLICIT}$ bu (in English *"What kind of woman $\langle$is$\rangle$ this"*)

  – Onu döven adam da şurada şu ağacın altındaki $\langle$(dir)$_{S-IMPLICIT}$ bu (in English *"The man who beat him $\langle$is$\rangle$ over there under that tree"*)

[22]The word "that" is omitted in the English corresponds to "diye" in Turkish.

- **Conjunctions and discourse connectives** In Turkish, conjunctions and discourse connectives are used to link two or more items that have the same syntactic function. While conjunctions are used to link phrases, subordinate clauses or sentences; discourse connectives are used to link sentences. Here we only provide the rules for conjunctions and discourse connectives, which are not covered in the other rules described above.
  - "**halbuki/oysa**": The course connectives "halbuki"/"oysa" (whereas/however) indicate a contrast between the two states. Since it is used to link different states, it is labeled with a `Linker`.

    Example:

    * $\langle$Oysa$\rangle_L$ benim sizlere ne kadar çok anlatacağım vardı (in English *"$\langle$However$\rangle$ I had so much to tell you"*)
  - "**peki**" It expresses the speaker's agreement on the subject. Therefore, it is labeled as `Ground`.

    Example:

    * $\langle$Peki$\rangle_G$ senin yerin neresi (in English *"$\langle$So$\rangle$ where is your place"*)
  - "**Demek**" It is used at the end or beginning of a sentence, and adds inferential meaning by referring to the previous sentence. Since it also contains the attitude of the speaker, it is labeled as `Ground`.

    Example:

    * Siz o dünyayı biliyorsunuz $\langle$demek$\rangle_G$ (in English *"$\langle$So$\rangle$ you know that world"*)
  - "**Yoksa**" It is an inferential connective and is used in yes/no questions. It is used when the speaker realizes that the situation is different from what s/he expects. It is labeled as `Adverbial`.

    Example:

    * $\langle$Yoksa$\rangle_D$ biliyor musunuz orayı diye hayretle sordu (in English *"Do you know where it is?" he asked amazedly"*)

- "**gibi**" The role of "*gibi*" is derived from its primary function as a postposition. It expresses the opinion of a speaker. Therefore it is labeled as `Ground`.

  Example:

  - Ne tuhaf başka şeyler önem kazandı $\langle$gibi$\rangle_G$ (in English *"How weird other things $\langle$seem$\rangle$ to matter"*)

- **Subordinators [137]:** A subordinator is a word or suffix that introduces a subordinate clause and connects it to the main clause or another clause.

  - **Simplex subordinators:** In Turkish "*-(y)Ip*" and "*-(y)ArAk*" are used as subordinating suffixes in simplex subordinators for nominal and adverbial clauses to link clauses. We labeled the clauses with "*-(y)Ip*" as `Parallel Scene` since they define a new Scene, whereas we labeled the clauses with "*-(y)ArAk*" as `D-Scene` since they modify the relation (Process or State) in the Scene.

    Examples:

    * $\langle$Dar yollarda $\langle$koşarak$\rangle_D$ giden$\rangle_D$ Kerem'i yakaladım (in English, *"I caught Kerem [$\langle$running$\rangle$ on narrow roads]"*)
    * $\langle$Kerem çayına iki şeker $\langle$atıp$\rangle_P\rangle_H$ [yavaşça karıştırdı$]_H$ (in English, *"[Kerem $\langle$added$\rangle$ two sugars to his tea] and [stirred it slowly]"*)
    * $\langle$Yataktan $\langle$kalkıp$\rangle_P\rangle_H$ $\langle$balkona çıkmış$\rangle_H$ (in English, *"[S/he $\langle$got out of$\rangle$ the bed] and [went out on the balcony]"*)

  - **Complex Subordinators** They consist of a postposition and a nominalizing suffix. We labeled the postposition in the complex subordinators as a `Linker` since it connects the adverbial clause evoking a Scene with the main Scene.

    Examples:

    * $\langle$Ona yetişebilmek$_P\rangle_H$ $\langle$için$\rangle_L$ $\langle$peşinden koşuyordum$\rangle_H$ (in English, *"I was running after her/him $\langle$to$\rangle$ catch up"*)
    * $\langle$Bir şey anlatmak$_P\rangle_H$ $\langle$için$\rangle_L$ $\langle$[gelmiştin buraya$\rangle_H$ (in English, *"You came here $\langle$to$\rangle$ say something"*)

$*$ $\langle$O özgürlüğünü teslim ettiği$_P\rangle_H$ $\langle$için$\rangle_L$ $\langle$sanki rahatlamıştı$\rangle_H$ (in English, *"It was as if she was relieved* $\langle$*that*$\rangle$ *she had surrendered her freedom"*)

5.3. Inter-annotator Agreement

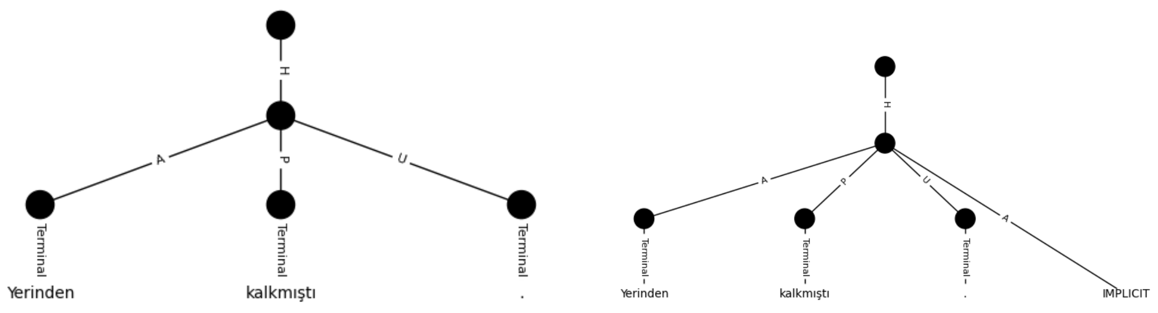
We employed 2 annotators who are native Turkish speakers with expertise and knowledge in computational linguistics. The annotators were initially trained for UCCA annotation based on the official UCCA guideline [17]. The annotation step was performed individually, which was followed by a comparison phase in which the two annotations were compared to each other to identify annotation agreements. To show how clear our annotation guideline is and how uniformly it was understood by the annotators, we used accuracy and Cohen's kappa (κ) [138]. There was a disagreement in 616 edges out of $3,981$ edges in 400 sentences in total. Thus, the disagreement between the two annotators is 15.47% and Accuracy is 84.53% based on the annotated tokens. Accuracy does not consider the expected chance of agreements that are likely to occur. Therefore, we also calculated Cohen's kappa (κ), a statistical measure of the reliability of annotations between different annotators. Cohen's kappa (κ) is computed as follows:

$$kappa(\kappa) = \frac{P(A) - P(E)}{1 - P(E)} \tag{13}$$

where $P(A)$ is the agreement between annotators, identical to accuracy, and $P(E)$ is the probability of chance agreement. Cohen's kappa (κ)(*100) score for the annotation is 82.29. The results between 80 and 90, indicate a strong agreement between the annotators. The general disagreement that recurs in the training procedure mainly concerns the annotation of the clitics.

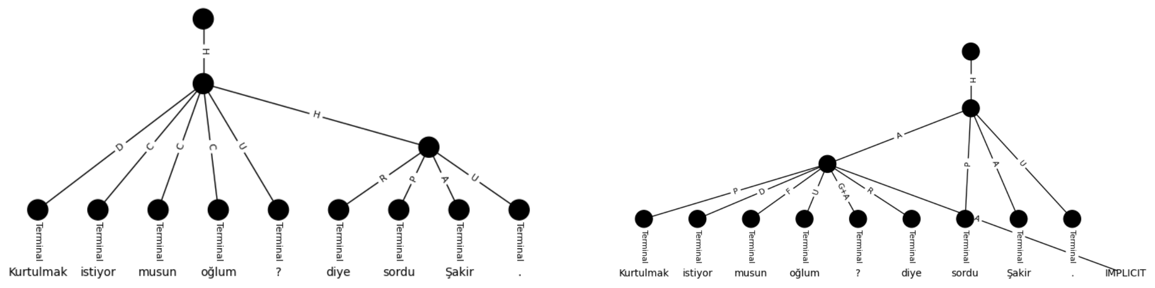
5.4. Comparison of the Outputs in Automatic and Manual Annotation

We analyse the outputs obtained from the automatic annotation to see what kind of errors are mostly corrected during manual annotation. Two example sentences parsed with the external semantic parser are given in Figure 5.2 and 5.3 along with their gold annotations. For the



(a) The output of the semantic parsing model (b) The gold annotation of the sentence

Figure 5.2 The semantic parse tree obtained from the semantic parsing model and the gold annotation obtained from the manual annotation of the sentence, *Yerinden kalkmıştı.* (in English, *"S/he had stood up."*)



(a) The output of the semantic parsing model (b) The gold annotation of the sentence

Figure 5.3 The semantic parse tree obtained from the semantic parsing model and the gold annotation obtained from the manual annotation of the sentence, *Kurtulmak istiyor musun oğlum? diye sordu Şakir.* (in English, *"Do you want to be saved son? asked Şakir."*)

first sentence in Figure 5.2, we added only the `IMPLICIT` edge. However, in the second sentence given in Figure 5.3, the annotation of the labels was mostly incorrect and had to be corrected manually.

When correcting the annotations obtained from the semantic parser model, we made almost no additional corrections for the short sentences (with less than 5 words). The labels of the terminal nodes were also mostly correct. We corrected most of the annotations for the `Parallel Scene` (H) and, for that matter, the entire annotation of the sentence.

	Turkish	En-Wiki	En-20K	Fr-20K	De-20K
# sentences	400	5,141	492	492	6,154
# tokens	2,474	158,739	12,638	13,021	144,529
# edges	3,981	208,937	16,803	17,520	187,533
% primary	96.66	97.40	96.79	97.02	97.32
% remote	3.34	2.60	3.21	2.98	2.68
% Participant (A)	26.78	17.17	18.1	17.08	19.86
% Center (C)	10.02	18.74	16.31	18.03	14.32
% Adverbial (D)	7.36	3.65	5.25	4.18	5.67
% Elaborator (E)	5.38	18.98	18.06	18.65	14.88
% Function (F)	2.41	3.38	3.58	2.58	2.98
% Ground (G)	1.41	0.03	0.56	0.37	0.57
% Parallel Scene (H)	13.06	6.02	6.3	6.15	7.54
% Linker (L)	0.68	2.19	2.66	2.57	2.49
% Connector (N)	0.00	1.26	0.93	0.84	0.65
% Process (P)	11.33	7.1	7.51	6.91	7.03
% Relator (R)	1.81	8.58	8.09	9.6	7.54
% State (S)	5.10	1.62	2.1	1.88	3.34
% Punctuation (U)	14.67	11.28	10.55	11.16	13.15

Table 5.1 Proportions of the edges and labels as well as the number of sentences and tokens in the UCCA datasets in Turkish, English, French, and German.

5.5. Annotation Statistics

We provide the statistical distributions in the annotated dataset with a comparison to other annotated datasets in English, German, and French (see Table 5.1). In particular, we provide the proportions of the edges and the labels in the final annotated dataset.

The average sentence length in the Turkish dataset is comparably shorter than that of the other languages. Turkish sentences are syntactically simple and involve one predicate and one subject. This results in fewer noun and verb phrases and consequently fewer Elaborator (E), Center (C), and Relator (R) labels in the Turkish dataset. Also, due to the simple sentences, the number of Linkers (L) is less than that of the other datasets. Due to the morphologically rich nature of the Turkish language, morphemes that convert a verb to a nominal or a verb into a verb are frequently used. Therefore, the number of Processes and States is higher compared to the other datasets.

5.6. Discussion

Due to the lack of annotated UCCA dataset in Turkish, one of the goals of this thesis is to build a Turkish UCCA dataset. Since the UCCA representation supports rapid annotation by non-experts who do not have proficient linguistic knowledge and does not require external features such as semantic role labels, PoS tags, or stems, it is important for AMR annotation. UCCA also enables parsing across languages using cross-lingual machine-learning approaches because it is applicable across languages. Therefore, we chose the UCCA representation as the graph-based semantic representation for the dataset and followed a semi-automatic annotation approach to reduce the annotation effort. We used the semantic parser proposed in the scope of the thesis for initial annotation and revised the annotated sentences for Turkish grammar rules. Unlike the French UCCA dataset annotated with an aligned corpus of *Twenty Thousand Leagues Under the Sea*, we used METU-Sabancı Turkish Treebank [103, 104] with a semi-automatic pipeline.

An important fact concerning the quality of the dataset. Therefore we calculated the accuracy and Cohen's kappa (κ) [138] as the agreement between the annotators. The accuracy and Cohen's kappa (κ)(*100) scores for the annotation are 84.53% and 82.29, respectively. The scores between 80 and 90 indicate a strong agreement between the annotators.

The guideline covers the specific Turkish grammar rules such as clitics, and subordinators in addition to the rules which are already defined in the actual UCCA guideline.

5.7. Summary

In this chapter, we presented the Turkish UCCA dataset with 400 sentences obtained from the METU-Sabanci Turkish Treebank. The annotation was performed in a semi-automatic framework in which we used an external semantic parser in zero-shot learning trained on UCCA datasets in other languages and tested with the raw Turkish dataset to obtain a partially annotated dataset. Then, we analysed the discrepancies between the annotated sentences and the English guideline to define new rules in line with Turkish grammar in addition to the ones which are already defined in the actual UCCA guideline. In doing so, we either utilized the current specifications by describing how each linguistic construction should be annotated to ensure consistent annotation based on the original guideline, or we defined new rules that cover the syntactic rules peculiar to Turkish.

We believe that this corpus will be a crucial resource for advancing the state of the art in semantic parsing in Turkish and especially in Turkish UCCA parsing. This will also be useful for other NLP tasks that require semantic information, such as QA, text summarization, and NMT.

Our contributions in this chapter are as follows:

- We presented the Turkish UCCA dataset with 400 sentences from the METU-Sabancı Turkish Treebank [103, 104].
- We developed a semi-automatic pipeline for the annotation process that can be applied to other languages.
- We prepared the Turkish UCCA guideline that covers the Turkish grammar rules that are not covered in the English guideline [135].

6. EXTRINSIC EVALUATION OF THE UCCA REPRESENTATION

In this chapter of the thesis, we utilize the UCCA framework for extrinsic evaluation, in which we apply UCCA to various NLP problems, including STS, text classification (binary, multi-class, and multi-label), and QA. Since the focus language of this thesis is Turkish, a low-resource language in terms of graph-based semantic representation data, we study Turkish. In addition to Turkish, we also study English, a rich-resource language.

In this chapter, we verify the effectiveness of UCCA representation on several NLP tasks and datasets. This chapter tries to answer the following research questions:

Research Questions:

1. Can semantic-aware models that use the UCCA representation perform relatively well on NLP tasks compared to other current SOTA methods that do not utilise any semantic information?
2. Are there differences in terms of the performance between semantic-aware and syntax-aware models on NLP tasks?
3. Do semantic-aware models with different graph-based semantic representations achieve different results on NLP problems?
4. Do the semantic-aware models using graph-based semantic representations achieve different results in NLP problems? Is the NLP problem important for the performance of the semantic-aware models?

We first present the proposed models with different representations (semantic, syntactic). Then, we discuss the results of the proposed models with different representations for several NLP problems to answer the questions posed above.

6.1. Related Work

6.1.1. Semantic Textual Similarity

The identification of STS in short texts was proposed in 2006 [139, 140], where the goal was to identify whether two text segments are paraphrases of each other or not. Between 2012 and 2017, the semantic similarity task has been one of the main tasks in SemEval [141–145] and the proposed models based on neural networks [146–150] were not only able to identify a similarity between two texts but also able to generate a similarity score (usually between 0 and 5).

Measuring semantic similarity between texts has been performed using several methods in the literature [151]: (i) the topological method, which utilises external semantic resources such as WordNet in order to assess the similarity between two texts based on the topological distance on such semantic networks [152–156], (ii) statistical similarity, which exploits mainly statistical vector-based models along with dimension reduction techniques to assess the similarity between two texts [153, 157, 158] , (iii) semantic-based method, which combines a set of similarity measures such as soft cardinality [159], word n-gram overlap to predict similarity between texts [148, 160], symbolic regression [161], (iv) ML method, which builds a mathematical model based on lexical, syntactic and semantic features to compute the similarity between given texts [3, 4, 150, 162–166].

Semantic similarity methods have recently made the most out of recent developments in neural networks, especially the recent neural word embedding approaches. The most commonly used neural network architectures for semantic similarity are Convolutional Neural Networks (CNN) [150, 167], Long Short Term Memory Networks (LSTM) [168], Bidirectional Long Short Term Memory (BiLSTM) [169], Recursive Tree LSTMs [170], and Decomposable Attention Model (DAM) using n-grams [171].

With state-of-the-art results on sentence-pair regression-classification tasks such as natural language inference [172] and QA [173, 174], obtained from BERT [91], the pre-trained language models are also applied to the STS task [4, 175–178].

In addition to supervised models, unsupervised models are also used with contrastive learning with positive and negative sentence pairs. In one of the studies, SimCSE [3] uses different dropout masks to compute different sentence vectors from an input sentence, and ConSERT [179] uses token shuffling and adversarial attacks. The problem of the data augmentation methods used in SimCSE [3] and ConSERT [179] is the meaning changes and the generation of dissimilar positive pairs as a result of the meaning changes. DiffCSE [166] uses masked language modeling based on word replacement instead of data augmentation. DebiadCSE [164] designs an instance weighting method to penalize false negatives in the model.

Semantic similarity has also been used in other tasks such as recommendation systems [180], and code clone detection [181].

6.1.2. Text Classification

Text classification is a classical problem in NLP with a wide range of applications including sentiment analysis [182], content detection [183], emotion classification [10]. With the growing scale of data in industrial applications, neural approaches have been applied to text classification problems, such as base DL models [183, 184], transfer learning models [185, 186].

We present related work on text classification problems that we study in the scope of the thesis.

6.1.2.1. Irony Detection Irony detection is becoming more popular as social media data increases. Yet, it is challenging for both humans and automated tools. This motivates the NLP community to solve the irony detection task. To raise awareness of irony detection, several shared tasks for different languages have been organized in the last decade, including SemEval [187], IronITA [188], IroSVA [189], and DEFT [190].

Neural approaches stand out in the irony detection task with their high performance as in other NLP tasks. Wu et al. [184] took the first place with a neural approach based on a densely connected LSTM augmented with pre-trained word embeddings, sentiment, and PoS tag features in Semeval 2018 for English [187]. Similar to Wu et al. [184], Cimino et al. [191] proposed a BiLSTM model in which the last hidden layer of the model is concatenated with sentiment features and sentence embedding features generated via the TweetToLexiconFeatureVector [192] and TweetToSentiStrengthFeatureVector [193] libraries. At DEFT 2017 [190], the winning team [194] proposed an approach based on a CNN model combining 3 different sentiment-based embeddings trained with word2vec and BERT, which are context, target, and sentence context embeddings in French. Last but not least, González [189] used a Transformer-Encoder model for the same task in Spanish.

While the irony detection task has been tackled well in high-resource languages with publicly available datasets, there is limited work conducted on Turkish, which is a low-resource language when it comes to new NLP tasks such as irony detection. The studies conducted on the Turkish irony detection task focus on datasets based on social media data [8, 9]. The first study on irony detection in Turkish was conducted by Taslıoglu and Karagoz [195]. The authors built a new dataset from Twitter and analyzed the performance of a set of supervised learning algorithms including Naive Bayes, Support Vector Machine (SVM), and k-Nearest Neighbors (kNN). They incorporated textual features that are commonly used in irony detection problems in the literature such as bracketed question marks, exclamation marks, smileys, and diminutive forms. Similar to the study of Taslıoglu and Karagoz [195], the study of Dülger [196] applied mostly traditional ML algorithms such as C4.5, Naive Bayes, Logistic Regression (LR), as well as MLP. They used features based on patterns and polarity. In the studies of Taslıoglu and Karagoz [195] and Dülger [196], irony detection was treated as a binary classification problem and ML algorithms were compared using various features. Cidecio et al. [8] compared DL approaches such as LSTM, BiLSTM, and BERT for irony detection using a new dataset from Twitter. There were a few complications in their methodology due to the small size of the dataset, character limitation on Twitter, grammar errors, and emojis. DL approaches were more successful compared to ML models as demonstrated in the study of Cidecio et al. [8]. Ozturk et al. [9] extended the study of Cidecio et al. [8] by increasing the size of the dataset and compared the DL approaches with traditional ML models. The results showed that DL models perform better than traditional ML models. They also analyzed the effect of using polarity scores and graph-based features in traditional ML models. The results obtained with graph-based features are higher than that of other models.

6.1.2.2. Multi-label Emotion Classification Emotion detection has been studied in recent years [197–199], and with the success of DL models in NLP tasks, such as NMT [23, 200], STS [162, 163], the advanced DL models also employed in solving the MLEC problem, such as LSTM [201, 202], CNN [203], GRU [204, 205], Transformers [206–208]. The MLEC problem is a popular task that is also addressed in the SemEval-2018 shared task [10] for English, Arabic, and Spanish. The SemEval shared tasks released for emotion classification play an important role in developing resources for emotion classification [201, 209].

NTUA-SLP [201], which took first place in the emotion classification subtask of the SemEval-2018 shared task [10], presented BiLSTM with multi-layer self-attention mechanism. As pre-trained embedding, they trained word2vec [210] with 550 million English tweets, augmented with a set of effective word embeddings trained with the word embeddings. To fine-tune the hyperparameters of the proposed model, the authors adopted a Bayesian optimization approach that allows a time-efficient search for all possible values in the high-dimensional space instead of a grid or random search. One of the other teams that participated in SemEval-2018 is TCS Research [211], which combined three different features (lexicon, DL features extracted from BiLSTM, SentiNeuron [212] features) into a SVM to develop a unified ensemble system. The system was designed to handle noisy sentiment multi-label datasets with a mixture of embeddings in parallel. The team that placed third in the SemEval shared task [10], PlusEmo2Vec, applied neural network models to extract features and used these features in traditional ML classifiers (logistic regression, and support vector regression (SVR)). They also extended the training set by using an external dataset provided by the competition to learn a better representation of emojis and #hashtags. The other studies also applied DL networks such as MLP [213], LSTM [202].

In addition to the SemEval shared task [10], multi-label emotion detection [6, 214, 215] is a well-studied problem by researchers using various datasets such as GoEmotions [6]. Demsky et al. [6] presented the GoEmotions dataset, a large, manually annotated dataset collected from Reddit comments. The authors also presented baseline scores using the pre-trained language model BERT [91].

6.1.2.3. Transfer Downstream Tasks Learning sentence embeddings is a fundamental problem in NLP and has been extensively studied in the literature [5, 163, 216]. To evaluate the quality of sentence embeddings, the SentEval toolkit[217] was introduced, which uses a variety of tasks, including binary and multi-class classification.

6.1.3. Question Answering

Question Answering (QA) is a field of Information Retrieval (IR) and NLP that builds systems to automatically answer questions posed by humans in natural language. The research field of QA is very diverse, e.g., due to the type of the questions (multiple choice, conversational, visual), the type of the answers (factoid, definition-based, hybrid), the source of answer evidence (hybrid, raw text, knowledge base), the modeling approach for answer retrieval (rule-based, ML, DL) [218].

Here we focus on factoid question answering, also called Reading Comprehension (RC), since an understanding of natural language and knowledge about the world is required to answer the question. One of the well-studied datasets for this problem is SQuAD (Stanford Question Answering Dataset) [219]. Prior to the success of PLMs in NLP tasks, studies focused on attentional mechanisms including Attention Sum [220], Self-matching [200], and Attention over Attention [221] for this problem. With the success of PLMs such as BERT [91], ALBERT [222], and ELECTRA [223] show a strong capacity to capture the contextualized sentence-level language representations, which improves QA performance.

6.1.4. Syntax-aware Models

Syntax-aware models have been effectively used in various NLP problems, such as sentiment analysis [224, 225], text generation [226], QA [227], and SRL [227, 228], etc. In particular, dependency trees used as syntactic information in problems requiring the relationship between words such as NMT [229, 230], language model [12, 231], and SRL [232], are gaining popularity. The studies demonstrate that syntactic information integrated into the models improves the performance of the models.

Bastings et al. [229] presented Graph Convolutional Network (GCN) integrating syntax using dependency representations of the dataset for NMT. Nguyen et al. [233] proposed a hierarchical accumulation tree structure that integrates dependency trees into a self-attention mechanism, and Duan et al. [234] presented a neural method for data-augmentation using dependency trees for NMT.

Xie et al. [235] presented three methods by using parent-scaled self-attention (PASCAL), syntax-aware word representation (SAWR), or constituent attention (CA) to integrate dependency trees into the Transformer encoder of a seq2seq semantic parser to improve the performance of the semantic parser. Guo et al. [226] treated text generation as a graph generation problem that takes advantage of word-order and syntactic linkages. They developed word graph model using GNN. The method involved incremental sentence construction while maintaining syntactic integrity using a top-down, syntax-driven approach. Using a GNN, Schlichtkrull et al. [227] worked on QA and SRL problems. The authors developed a post-hoc technique for analyzing GNN predictions that highlights irrelevant edges. Marcheggiani and Titov [232] used GCN to encode the constituent structures and provide the information for the SRL system called SpanGCN.

One of the most important NLP tasks that integrate syntax information with dependency representations in the models is the language model. Li et al. [12] introduced syntax-aware local attention (SLA), which can be adapted to different tasks, and integrated SLA with BERT [91]. Huang et al. [224] used a graph attention network on a dependency tree structure and external pre-training knowledge from the BERT language model to better describe the relationship between context and aspect words. Gong et al. [236] presented syntax-graph guided self-attention (SGSA), which combines source-side syntactic knowledge with multi-head self-attention. Bugliarello and Okazaki [237] presented a method called PASCAL, which is a self-attention mechanism that integrates the dependency representation of the sentence. Sachan et al. [238] investigated fusion methods, late fusion and joint fusion, to combine syntax information from dependency trees in pre-trained transformers for NLP problems, namely SRL, NER, and RE.

6.1.5. Semantic-aware Models

In recent years, semantic-aware models have gained popularity due to their impressive performance on NLP problems such as classification [183, 239], RC [240, 241], text summarization [33, 242], language modelling [243], and NMT [25, 244–246]. The most commonly used semantic representation in the studies is the AMR graph-based semantic representation [183, 247, 248], which is relatively older than other representations.

Takase et al. [242] incorporated AMR into the standard encoder-decoder model using a modified version of tree-LSTM to improve the results of summarization. Dohare et al. [33] investigated a full-fledged text summarization pipeline with an AMR intermediate step. The pipeline first generated an AMR graph of the given input and extracted a summary graph of the generated AMR graph, and finally generated summary sentences from this summary graph. Liu et al. [34] proposed a statistical summarization method using AMR to parse sentences. The authors applied the summarization to AMR graphs and converted the graphs to text using AMR-to-text generator. Vlachos et al.[247] proposed an approach to guide NLP using particular information from an original text to generate its summary from its

AMR graph. The approach is based on a standard sequence-to-sequence model [249] for estimating a summary of an AMR graph. More specifically, information from an original text that is not present in an AMR graph is used to improve the quality of a summary. Jin et al. [250] proposed an encoder-decoder architecture that takes as input an original text along with its semantic graph to predict a summary. Kouris et al. [248] proposed a framework for generating summaries of individual documents using AMR semantic graph-to-text ML predictions. The ML phase of the proposed framework was performed in two steps: (1) tokens of the training set were represented in a continuous vector space (word2vec [210], BERT [91]), (2) vectors are fed into a DL model to predict a summary of an original text. The authors investigated five DL models: an attentive sequence-to-sequence model with a pointer generator, a reinforcement learning model, a transformer, a transformer with contextual embeddings, and a pre-trained encoder-transformer. Ribeiro et al. [251] presented FACTGRAPH, which decomposes documents and summary into a meaning representation with a graph encoder augmented with structure-aware adapters to capture interactions between concepts based on graph connectivity, in addition to text representations with an adapter-based text encoder.

Song et al. [23] showed that AMR graphs can be helpful in BiLSTM attention to NMT. Nguyen et al. [25] investigated the effect of AMR with other representations SRL and dependency trees in NMT models and proposed a method for incorporating AMR into NMT models such as Seq2Seq, ConvSeq2Seq, and Transformer. Li and Flanigan [246] presented a semantic-aware encoder-decoder architecture augmenting a Heterogeneous Graph Transformer (HGT) by Yao et al. [252] for the task of NMT. The proposed architecture combines graph representations of HGT with the semantic representation of AMR andthe sentence representation of Vanilla Sequence Transformer to produce a target sequence representation

Slobodkin et al. [245] proposed two parameter-free frameworks that integrate UCCA semantic representation into Transformers by semantics-aware masking of (some of) the attention heads for NMT. One of these systems acts on the encoder through a Scene-Aware Self-Attention (SASA) head and the other acts on the decoder through a Scene-Aware

Cross-Attention (SACrA) head. Zhang et al. [243] proposed a semantic BERT that combined BERT with SRL as a language model and applied it to NLU benchmark datasets that include natural language inference, RC, STS, and text classification.

Such a study to detect irony as a classification was proposed by Ahmed et al. [239], i.e., a study in which directed unweighted graphs were constructed for each tweet in the dataset and content similarity, maximum common subgraph similarity, and their variant were used to compare the graphs in terms of similarity and classify the tweets as ironic or not. the other classification problem toxic content detection, in which Elbasani and Kim [183] proposed a neural approach based on a CNN model that integrates AMR graphs with external semantic information.

Other NLP problems where semantic-aware models have been proposed are commonsense reasoning [253], sentence matching [254], and data augmentation [255].

It is seen that semantic-aware models are commonly applied to complex NLP problems that require abstract and semantic information about the text.

6.2. Semantic Textual Similarity

Semantic Textual Similarity (STS) deals with the similarity between two phrases or sentences which is the evaluation of sentences or phrases according to their degree of semantic similarity.

We propose a Siamese Recursive Neural Network (Siamese-RvNN), which is a combination of a Siamese Network [256] and a Recursive Neural Network (RvNN) [257] to learn sentence embeddings for the semantic similarity between two sentences in semantic textual similarity task.

The overview of the proposed model flow is shown in Figure 6.1. First, we use the self-attentive semantic parser model described in Chapter 4.2.1. trained with the UCCA dataset to extract UCCA representations of the sentences in the STS dataset. We use the

UCCA semantic representations of the STS dataset to train universal sentence embeddings and predict the similarity between sentence pairs for the STS task.

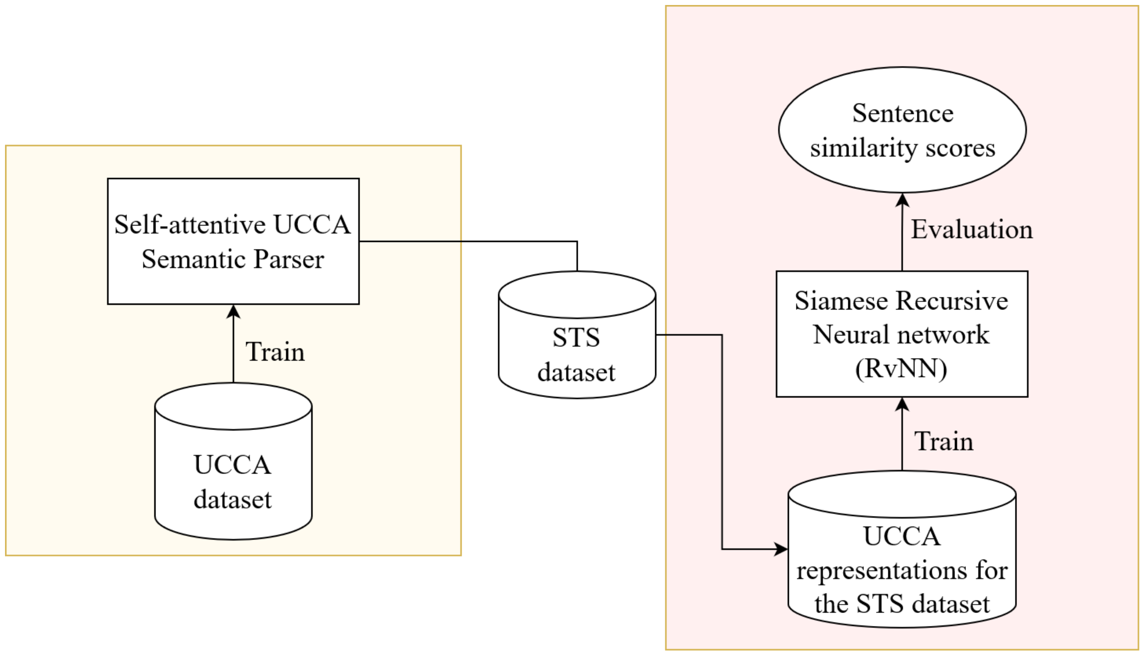
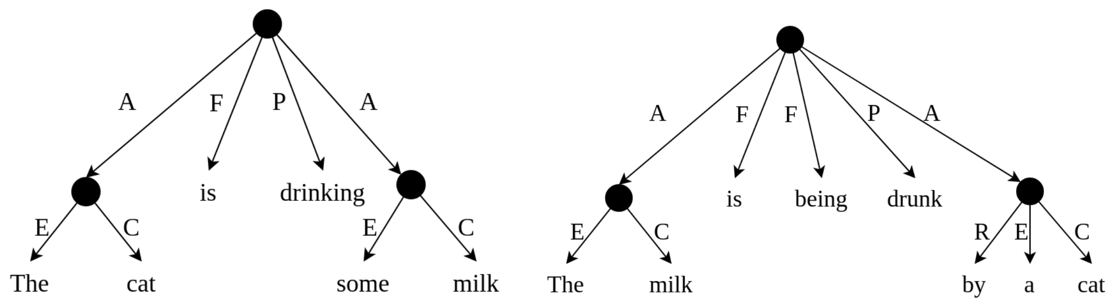

Self-attentive UCCA Semantic Parser
Train
UCCA dataset
STS dataset
Sentence similarity scores
Evaluation
Siamese Recursive Neural network (RvNN)
Train
UCCA representations for the STS dataset

Figure 6.1 The proposed system's overview for STS

6.2.1. Proposed Method

Recursive Neural Network (RvNN) [257] is a type of neural network that applies the same set of weights to a structured input. Each recursive network processes the nodes in topological order in the given structure (in the form of a graph or a tree) and recursively applies transformations to generate further representations from the previously computed representations of the children.

We build the RvNN model with UCCA representations constructed by the self-attentive UCCA semantic parser (see Section 4.2.1.) with a list of words represented as d-dimensional vectors in a pre-trained word embedding matrix $L \in \mathbb{R}^{d \times |V|}$ where $|V|$ is the size of the vocabulary. The UCCA graphs of the sentences in the STS dataset are obtained by the self-attentive semantic parser. The UCCA representations of an example pair of sentences taken from the STS dataset are given in Figure 6.2.



(a) The cat is drinking some milk (b) The milk is being drunk by a cat

Figure 6.2 UCCA-based semantic representations of a sentence pair taken from STS dataset

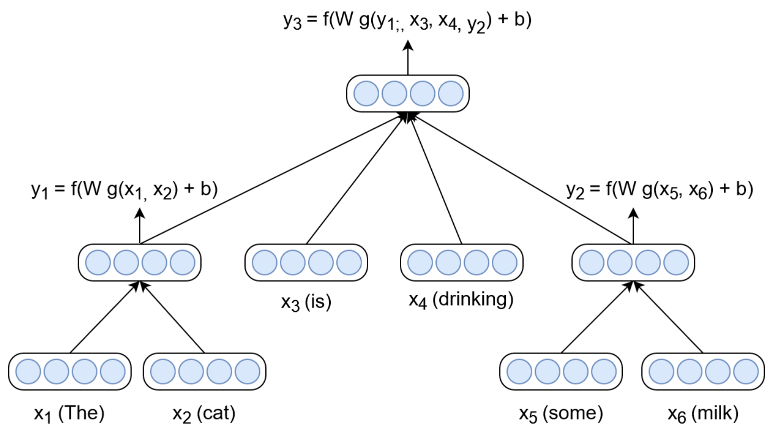


Figure 6.3 The composition process for “The cat is drinking some milk.” using RvNN

As illustrated in Figure 6.3, we obtain the representation of “The cat” by the composition of “The” and ”cat”, “some milk” by the composition of “some” and “milk” and the representation of “The cat is drinking some milk” is obtained by the vectors of “The cat”, “is”, “drinking” and “some milk”. The compositional sentence embedding is eventually generated based on the UCCA semantic representation of the sentence, which also gives semantic-informed sentence embeddings.

The composition is applied using a fully connected layer (i.e. one-layer MLP) for each node in the semantic graph. The mean of the input vectors is fed into the MLP since RvNN is adopted for non-binary trees that are the UCCA representation of sentences. For example, in Figure 6.3, x_1 <The> and x_2 <cat> are combined by the following nonlinear composition with weights W and the parent vector y_1 is computed and y_1, x_3 (<is>, x_4 <drinking> and

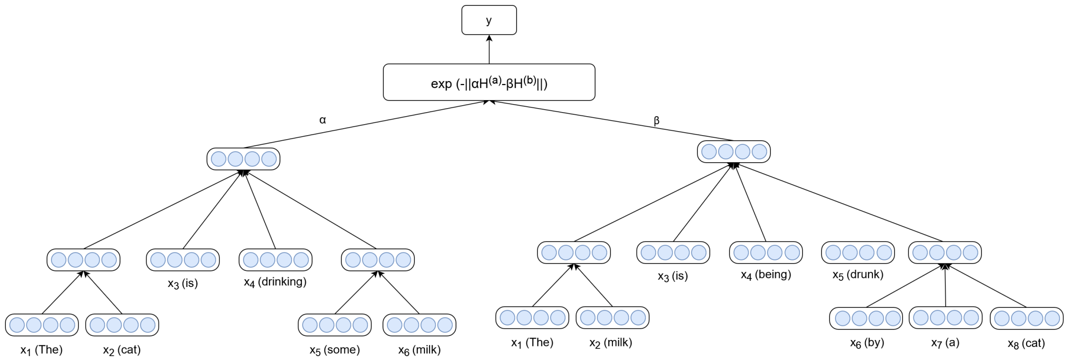


Figure 6.4 Overview of the Siamese-RvNN model architecture

y_2 are used to compute y_3 as follows:

$$y_1 = f(Wg(x_1, x_2) + b) \quad (14)$$

$$y_3 = f(Wg(y_1, x_3, x_4, y_2) + b) \quad (15)$$

where f is a nonlinear activation function ReLU, g is the representation extractor function that is mean in the model, W is the weight matrix (with a dimensionality of $d \times d$, where d is the embedding dimension of the pre-trained word embedding) and b corresponds to the bias vector in this layer. A single MLP is used for the model, so the weights are the same for all sentences in the dataset.

Here we combine Siamese Networks with RvNNs. Siamese networks [256] are dual-branch networks with bound weights. In other words, they are built on the same network, which is copied and merged with an energy function.

The Siamese architecture is given in Figure 6.4. There are two networks $RvNN_a$ and $RvNN_b$ that simultaneously process one of the sentences in a given sentence pair. An example pair of sentences is given in Figure 6.2. The training set consists of triplets (x_1, x_2, y), where x_1 and x_2 are sentences in a pair in the training set, and y is the similarity score that is between $[0, 5]$ and defines the semantic similarity between the two sentences. The goal is to minimize the distance between semantic similar sentences and maximize the distance between dissimilar sentences in the embedding space for each pair, which is followed during training.

We use the Manhattan distance [258] which performs comparatively better than other distance metrics in Recurrent Neural Networks (RNN) [259–261] to measure the similarity between sentences in a pair as follows:

$$g = exp(-\alpha H^{(a)} - \beta H^{(b)}) \in [0, 1] \tag{16}$$

Here, g is computed by the model where $H^{(a)}$ is the output of network $RvNN_a$ and $H^{(b)}$ is the output of network $RvNN_b$. α ve β are two parameters that are used to apply weighting on the output of the two $RvNN$ models: $H^{(a)}$ and $H^{(b)}$. We rescale the output to ensure that the similarity is in the range of $[0, 5]$.

6.2.2. Experiments & Results

6.2.2.1. Datasets We evaluated the Siamese-RvNN model on several STS tasks using the output of the semantic parser model. We evaluated on 7 datasets that provide labels between $[0, 5]$ that correspond to a degree of semantic similarity:

- **SICK Dataset [262]** is compiled for sentence level semantic similarity/relatedness task.
- **STS Datasets [141–145, 263]** involve 6 different datasets released by SemEval in years between 2012 to 2017 for the STS task.

We also evaluated the Siamese-RvNN model on 7 other transfer downstream tasks with given datasets:

- **Movie Review (MR) [13]** is a dataset annotated for sentiment classification task with 2 classes (binary classification).
- **Customer Review (CR) [264]** is a dataset annotated for product review classification task with 2 classes (binary classification).

Dataset	Train	Dev	Test	Task
MR	10,662	train in k-fold	test in k-fold	sentiment (movies)
CR	3,770	train in k-fold	test in k-fold	product reviews
SUBJ	10,000	train in k-fold	test in k-fold	subjectivity/objectivity
MPQA	10,606	train in k-fold	test in k-fold	opinion polarity
SST-2	67,349	872	1,821	sentiment (movies)
TREC	5,452	train in k-fold	500	question type
MRPC	4,726	train in k-fold	1,725	paraphrase

Table 6.1 The statistics of transfer downstream tasks

- **Subjectivity / Objectivity (SUBJ) [265]** is a dataset annotated for subjectivity objectivity classification task with 2 classes (binary classification).
- **Multi-Perspective Question and Answering (MPQA) [266]** is a dataset annotated for opinion polarity classification task with 2 classes (binary classification).
- **Stanford Sentiment Analysis 2 (SST-2) [267]** is a dataset annotated for sentiment classification task with 2 classes (binary classification).
- **Text Retrieval Conference (TREC) [11]** is a dataset annotated for question type classification task with 6 classes (multi-class classification).
- **The Microsoft Research Paraphrase Corpus (MRPC) [268]** is a dataset annotated for paraphrase detection task with 2 classes (binary classification).

The details of the datasets are given in Table 6.1.

6.2.2.2. Evaluation Metric We use the SentEval toolkit [217] to evaluate the results obtained from the STS task and use Spearman's rank correlation ρ as the evaluation metric [269]. We use the accuracy metric in all transfer downstream tasks and follow the same configurations defined in SentEval [23].

[23] `https://github.com/facebookresearch/SentEval`

6.2.2.3. Hyperparameters and Implementation Details For the STS task, we use a combination of SNLI [172] and MNLI [270] as in Reimers and Gurevych [4] and Gao et al. [3] to finetune embeddings. In all STS experiments, we assigned coefficients $\alpha = \beta = 1$. We use a batch size of 16, the Adam optimizer [108] for training, BERT [91] (bert-base-uncased) pre-trained embeddings, a dropout of 0.2, and a learning rate of $1e-4$ in the proposed Siamese-RvNN model.

We use semantic-informed sentence embeddings from the RvNN to train a logistic regression classifier for the transfer downstream tasks. In all transfer downstream experiments, 10-fold cross-validation is performed as used in the study of Reimers and Gurevych [4] and the SentEval toolkit [217].

We use the semantic parser described in Section 4.2.1. to extract the UCCA semantic graphs used in the Siamese-RvNN model. The parser is based on encoder/decoder architecture that solves the problem as chart-based constituency parsing. The encoder of the architecture is the self-attention layers of the Transformer proposed by Vaswani et al. [92] with 2 MLP classifiers with 2 fully-connected layers and ReLU activation function and the decoder is the CYK algorithm [94]. The model is trained using the SemEval 2019 shared task dataset [1], which includes English, German, and French languages. We use the cross-lingual experiments of the model described in Section 4.3..

6.2.2.4. STS Task Results We conduct experiments on 7 STS tasks and report the performance of the proposed method in Table 6.2. The Siamese-RvNN model significantly improves the results on all datasets except SICK-R. Our model outperforms the previous best average Spearman's correlation with an improvement from 83.76 to 83.98, indicating that semantic annotation with UCCA helps to learn better sentence embeddings than other models such as SBERT [4] that uses pre-trained BERT along with Siamese and triple networks, and SimCSE [3], a simple contrastive sentence embedding framework that uses pre-trained BERT [91] and ROBERTA [111] with an MLP layer that can generate sentence embeddings from either unlabeled or labeled data. These two models use only pre-trained language models to capture sentence embeddings without using any semantic structure of the text.

Our proposed model can significantly improve the results on 6 out of 7 datasets and outperforms the previous state-of-the-art.

	Our Proposed Model							
Model	STS12	STS13	STS14	STS15	STS16	STS-B	SICK-R	Avg.
Siamese-RvNN	**78.46**	**87.75**	**82.61**	**86.83**	**84.39**	**87.03**	80.82	**83.98**
	State-of-the-art							
Universal Sentence Encoder ♣	64.49	67.80	64.61	76.83	73.18	74.92	76.69	71.22
SimCSE-BERT$_{base}$ ♠	75.30	84.67	80.19	85.40	80.82	84.25	80.39	81.57
SimCSE-Roberta$_{large}$ ♠	77.46	87.27	82.36	86.66	83.93	86.70	**81.95**	83.76
SBERT-NLI-large ♡	72.27	78.46	74.90	80.99	76.25	79.23	73.75	76.55
SRoBERTa-NLI-large ♡	74.53	77.00	73.18	81.85	76.82	79.10	74.29	76.68

Table 6.2 Task performance on STS tasks (Spearman's correlation X 100). (♣ results from [2]; ♠ results from [3]; ♡ results from [4])

6.2.2.5. Transfer Downstream Task Results Table 6.3 shows the evaluation results of the transfer downstream tasks. Siamese-RvNN achieves the best performance on 2 of 7 tasks. Although we were not able to outperform the state-of-the-art results on average, we generally achieved competitive results compared to other methods.

Model	MR	CR	SUBJ	MPQA	SST-2	TREC	MRPC	Avg.
Siamese-RvNN	84.25	89.54	**94.68**	89.10	**91.25**	88.42	74.93	87.45
BiLSTM ♢	81.1	86.3	92.4	90.2	-	-	-	-
Universal Sentence Encoder ♣	80.09	85.19	93.98	86.70	86.38	93.2	70.14	85.10
SimCSE-BERT$_{base}$ ♠	83.64	89.43	94.39	89.86	88.96	**89.60**	**76.00**	87.41
SBERT-NLI-large ♡	**84.88**	**90.07**	94.52	**90.33**	90.66	87.4	75.94	**87.69**

Table 6.3 Transfer downstream test results of the proposed model (measured as accuracy). (♣ results from [2]; ♠ results from [3]; ♡ results from [4]; ♢ results from [5])

6.3. Text Classification

Text classification is one of the most important tasks that aims to assign a sentence or document to one or more predefined classes. Text classification spans a variety of tasks such as emotion classification, sentiment analysis, news classification, etc.

We propose Graph Neural Networks (GNN) integrated syntactic or semantic information as input to different types of text classification problems, namely irony detection (binary classification), STS downstream tasks (binary and multi-class classification), and emotion classification (multi-label classification).

The overview of the workflow is shown in Figure 6.5. First, we use the UCCA/AMR semantic parser models and the dependency parser to extract semantic and syntactic representations of the sentences in the text classification datasets. Then we use the semantic and syntactic representations in GNN models for the text classification task.

6.3.1. Proposed Methods

6.3.1.1. Graph Neural Network For over a decade, Graph Neural Network (GNN) has been used to process graph data in various fields [271], for example, in computer vision, where images can be viewed as graphs with a regular structure and each pixel represents a node connected to adjacent pixels by an edge [272, 273]. GNN has also been used in NLP, where a text can be digitized by assigning indices to each character or word, resulting in a simple graph where each token is a node and is connected to another node by an edge [274, 275].

There are several approaches to transforming a text into a graph, such as digitizing text [274], statistical methods (PMI, TF-IDF) [276], dependency trees [277] or semantic graphs (AMR) [183].

6.3.1.2. Preprocessing We propose GNNs that integrate semantic or syntactic information. For this purpose, we use the adjacency matrix and the feature matrix extracted from UCCA/AMR graph-based semantic representations, as well as dependency trees as input to the models. We can divide the preprocessing into two steps: (1) converting the datasets into graphs/trees for each problem and (2) extracting the adjacency and feature matrices from the graphs/trees.

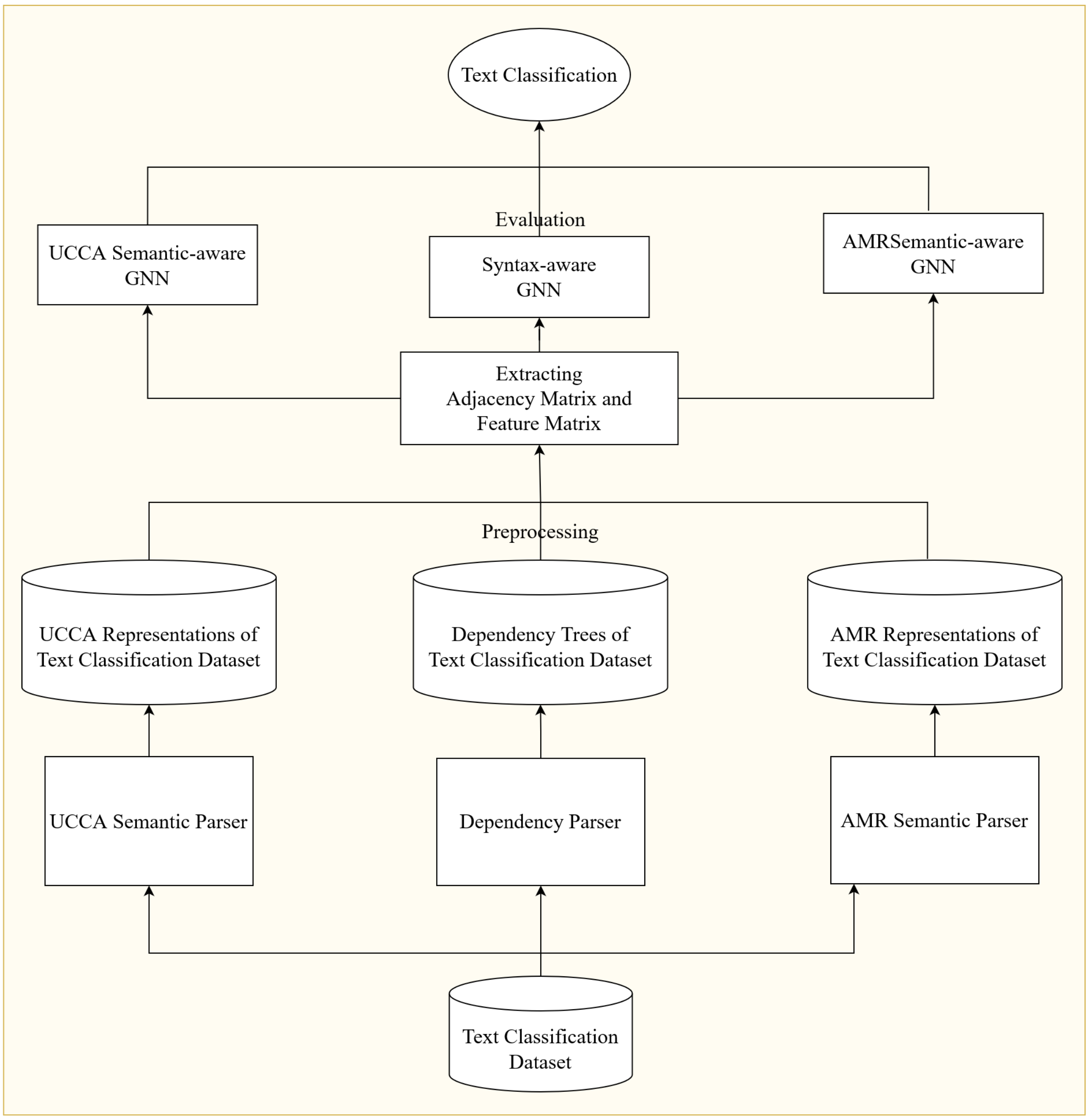
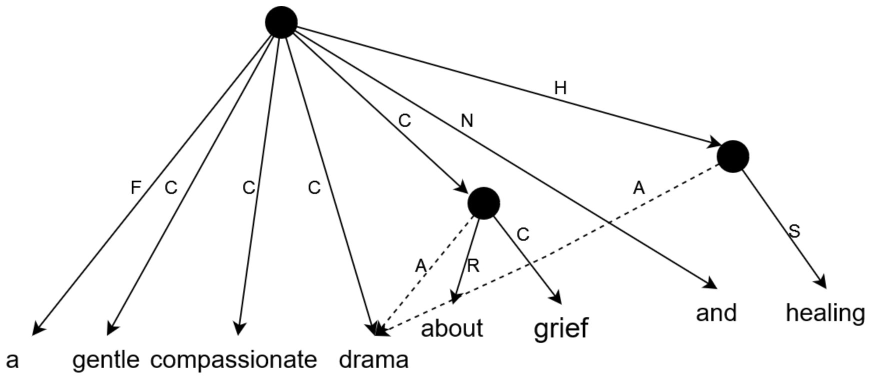

Text Classification
Evaluation
UCCA Semantic-aware GNN
Syntax-aware GNN
AMRSemantic-aware GNN
Extracting Adjacency Matrix and Feature Matrix
Preprocessing
UCCA Representations of Text Classification Dataset
Dependency Trees of Text Classification Dataset
AMR Representations of Text Classification Dataset
UCCA Semantic Parser
Dependency Parser
AMR Semantic Parser
Text Classification Dataset

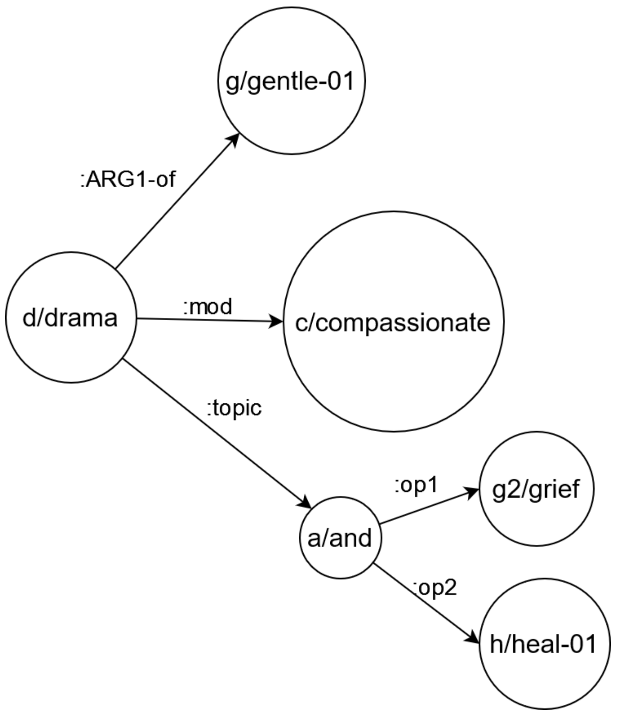
Figure 6.5 The proposed system's overview for text classification

1. **Converting datasets into graphs/trees** The parser models we used to extract graphs and trees from the datasets are listed below:

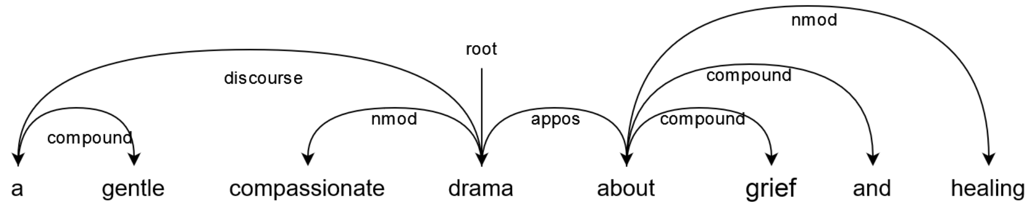
   - **UCCA Semantic Parser** We use the semantic parser described in Section 4.2.1. to extract the UCCA semantic graphs. The parser is based on an encoder/decoder architecture that tackles the parsing problem as chart-based constituency parsing. Self-attention layers of the transformer [92] are used with 2 MLP classifiers with 2 fully-connected layers and a ReLU activation function in the output layers in



(a) UCCA semantic representation



(b) AMR semantic representation



(c) Dependency representation

Figure 6.6 The semantic and syntactic representations of the Tweet *"a gentle compassionate drama about grief and healing"* taken from the MR dataset [13] obtained from the parser models, i.e., the UCCA semantic, AMR semantic, and dependency parsers.

the encoder. The output layers of the encoder generate the per-span scores where spans correspond to the constituents in the constituency tree. In the decoder, the CYK algorithm [94] is used to generate a constituency tree with the maximum score using the per-span scores obtained from the encoder. The semantic parser extracts the UCCA representation of each tweet in the datasets.

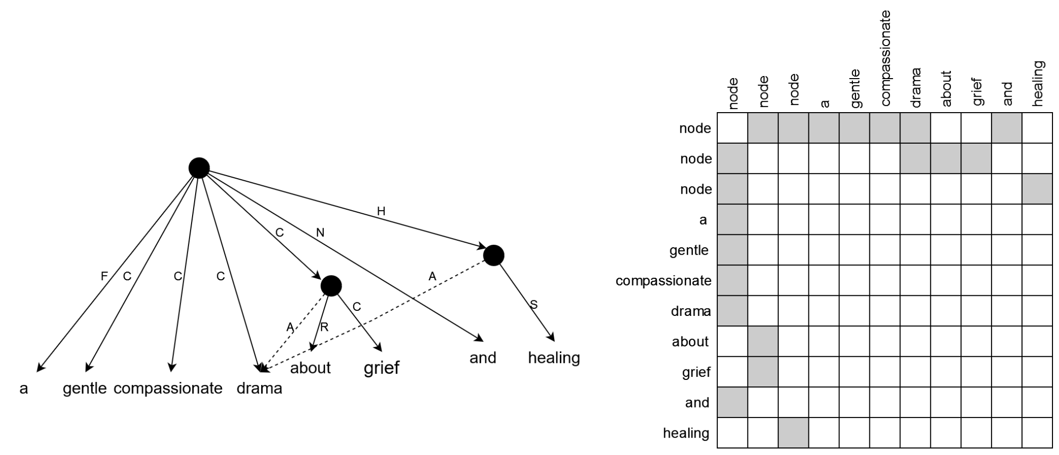
- **AMR Semantic Parser** We adopt an external semantic parser for AMR proposed by Roberts et al. [278] called T5 is a language model that has been finetuned on English sentences and their AMRs. We use AMRLib integrated[24] into spaCy library [279][25] to extract the AMR graphs of the datasets.
- **Dependency Parser** We use an external dependency parser proposed by Dozat and Manning [280] called Deep Biaffine Neural Dependency Parser to extract the syntactic representation of the datasets for the problem. The parser model follows the BiLSTM model with biaffine classifiers to predict arcs and labels. We use the model[26] integrated into the Stanza library [281] to extract the dependency trees of the datasets.

Example of UCCA, AMR, dependency tree representations of *"a gentle compassionate drama about grief and healing"* taken from the MR dataset [13] is given in Figure 6.6.

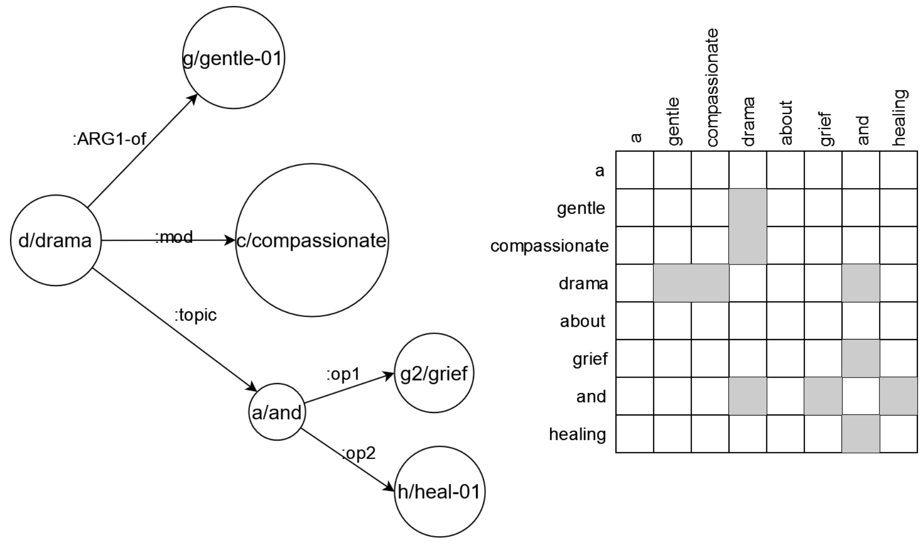
2. **Extracting adjacency and features matrices from graphs and trees** Since the input of the model, we use adjacency and features matrices from graphs/trees.

   Since the inputs of the proposed model are adjacency and feature matrices, we extracted the matrices from graphs and trees. The semantic representations of UCCA and AMR are based on DAG, and the dependency trees are represented by trees. We followed the same procedure, assuming that the dependency tree is a graph, so we could use the same procedure for dependency trees

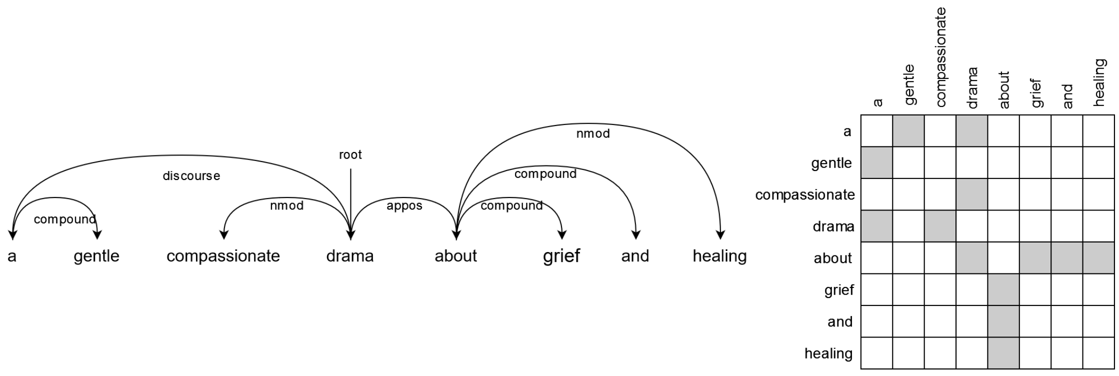
[24]`https://amrlib.readthedocs.io/en/latest/`
[25]`https://spacy.io/universe/project/amrlib`
[26]`https://stanfordnlp.github.io/stanza/depparse.html`



(a) UCCA semantic adjacency matrix



(b) AMR semantic adjacency matrix



(c) Dependency adjacency matrix

Figure 6.7 The adjacency matrix extracted from UCCA, AMR, and dependency representations of the sample *"a gentle compassionate drama about grief and healing"* taken from MR dataset [13]. The gray color in the matrix represents the value of 1 and the white color to the value of 0.

For a given graph $G = (V, E)$, where V is the set of nodes and E is the set of labeled edges (UCCA - edges, AMR - relations between nodes, dependency tree - dependency relations). We extracted the feature matrix X ($n \times k$, where n is the number of nodes (UCCA - terminal and non-terminal nodes, AMR - words, dependency tree - words skipping the node `ROOT`) in the graph and k is the embedding dimension) and the adjacency matrix A ($n \times n$, where n is the number of nodes in the graph). For the feature matrix, we used pre-trained word embeddings (BERT [91], RoBERTa [111], etc.) for nodes (UCCA - terminal nodes, AMR - words, dependency tree - words) and a randomly generated embedding with the same embedding dimension of the pre-trained word embeddings for non-terminal nodes in UCCA.

The overview of the approach is illustrated in Figure 6.8 for UCCA/AMR semantic graphs and dependency trees.

6.3.1.3. Models We use three types of GNNs in this study: Convolutional Neural Networks (CNN), Graph Convolutional Networks (GCN), and Graph Attention Networks (GAT).

- **Convolutional Neural Network**

  Convolutional Neural Networks (CNN) are inspired by the visual cortex of the animal brain. Recent work shows that CNNs perform very well in text classification [282, 283]. CNNs are used similarly in both text and image classification. The only difference is that text classification uses a matrix of word vectors instead of pixel values.

  In this study, we extend the CNN model by incorporating semantic graphs and dependency trees in the embedding layer. In order to incorporate the semantic graphs and dependency trees, we use the feature matrix of each sample in the dataset obtained from the external semantic or dependency parsers. The architecture of the CNN model is given in Figure 6.9.

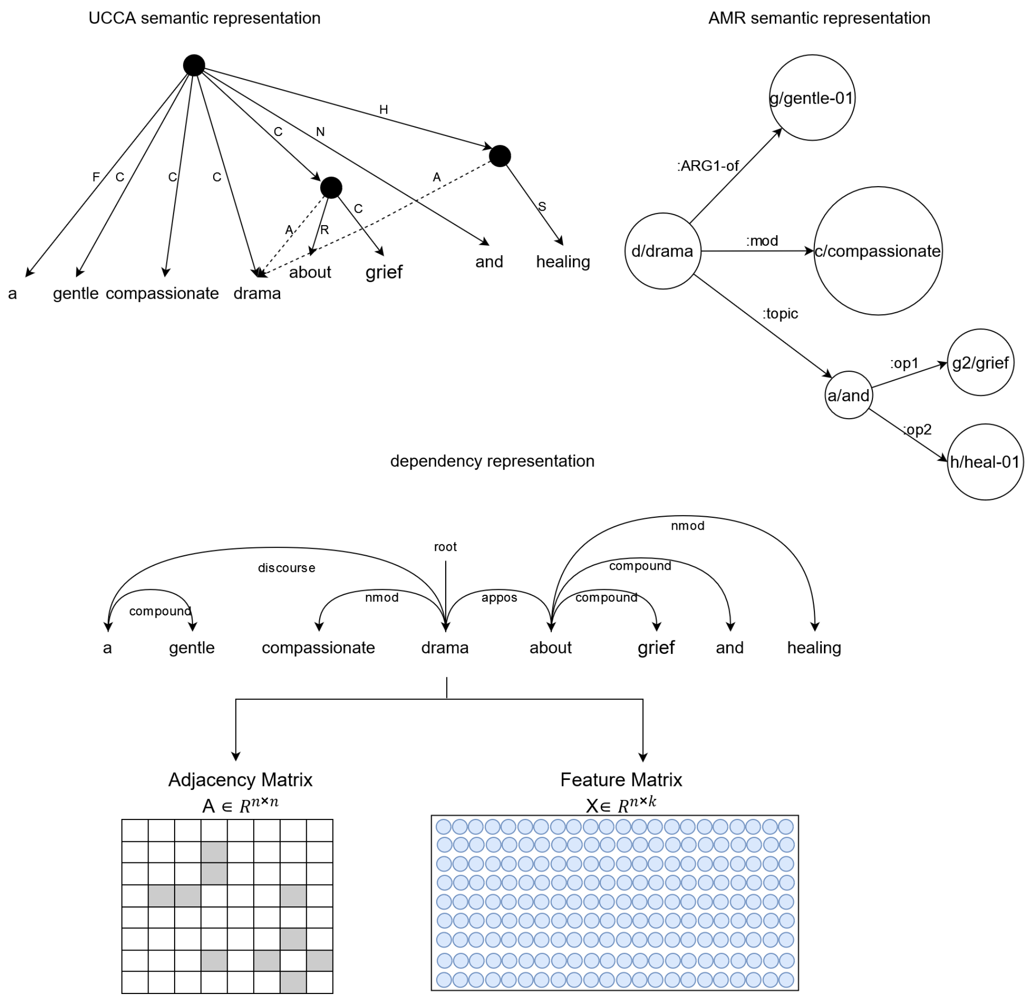


Figure 6.8 The overview of the extraction of the UCCA, AMR semantic graphs, and dependency trees along with the feature and adjacency matrix that will be integrated into the GNNs. Each row in the feature matrix corresponds to the pre-trained word embedding of a node in the graph/tree.

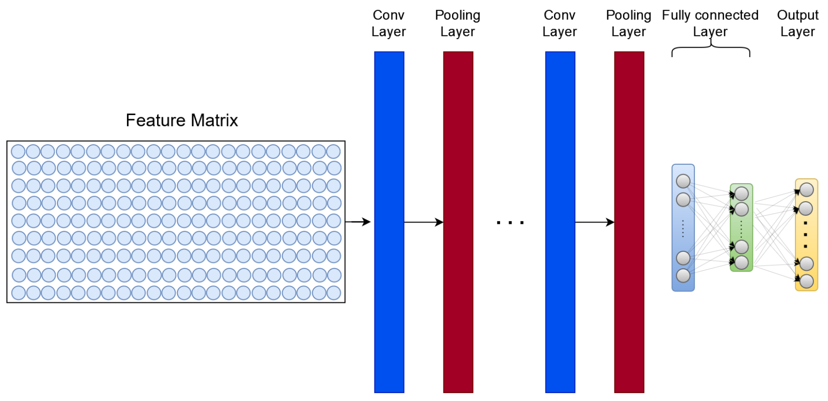


Figure 6.9 Architecture of CNN model for text classification

As can be seen in Figure 6.9, the embedding layer of the CNN model consists of feature matrices that are extracted from the representations and map the representations into low-dimensional vectors. The feature matrix in the embedding layer is fed into the convolution layer. Predefined filters roll over the feature matrix and reduce it into a low-dimensional matrix. The next layer is the pooling layer, where max-pooling is applied. In the fully connected layer, the output of the last pooling layer is fed into the linear layer. Since we applied the CNN model for the binary classification problem (irony detection), we use a softmax layer as the output layer to learn the prediction for the given sample:

$$Z = softmax(H^l) \tag{17}$$

where H^l is the final fully connected layer.

- **Graph Convolutional Network**

  In the first text classification model, which uses GCNs [276], each document is represented by a GCN and the embeddings of the nodes are induced based on the properties of their neighborhoods. We adopt their approach for the text classification problem. For this purpose, we use the adjacency matrix and the feature matrix extracted from the representations (UCCA/AMR/dependency) of each sample. The architecture of the GCN model is shown in Figure 6.10.

  The hidden layer of the GCN is computed as follows:

$$H^{i+1} = \sigma(A \cdot H^i \cdot W^i) \tag{18}$$

  where Wi is the weight matrix for layer i, A is the adjacency matrix, H^i is the feature matrix of the hidden layer ($H^0 = X$, where X is the feature matrix extracted from the representations, and σ is ReLU non-linear activation function.

  In the proposed model, we apply a multi-layer GCN where the layer size is a hyperparameter that needs to be tuned in the graph.

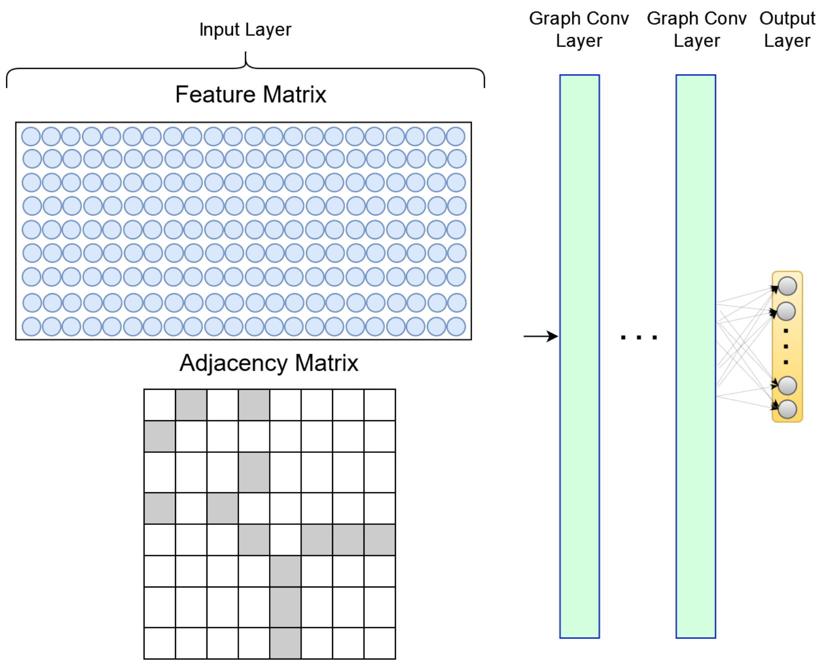


Figure 6.10 Architecture of the GCN model for text classification

Similar to the CNN model, we applied the GCN model for the binary classification problem (irony detection). We feed the output of the node in the final layer into the softmax layer to learn the class of the given text for binary and multi-class classification:

$$Z = softmax(H^l) \tag{19}$$

where H^l is the feature matrix of the final GCN layer.

- **Graph Attention Network**

  Graph Attention Network (GAT) is proposed by [284], which applies self-attention layers into GAT to address the shortcomings of the GCN model which assigns equal importance to every neighbor. GATs have been previously used in sentiment analysis by incorporating dependency parsing trees [176, 285, 286]. We extend the GATs by incorporating semantic graphs or dependency trees in the embedding layer. We use the adjacency matrix and the feature matrix extracted from the representations (UCCA/AMR/dependency) for each sample in the dataset, similar to the GCN model. The architecture of the GAT model is given in Figure 6.11.

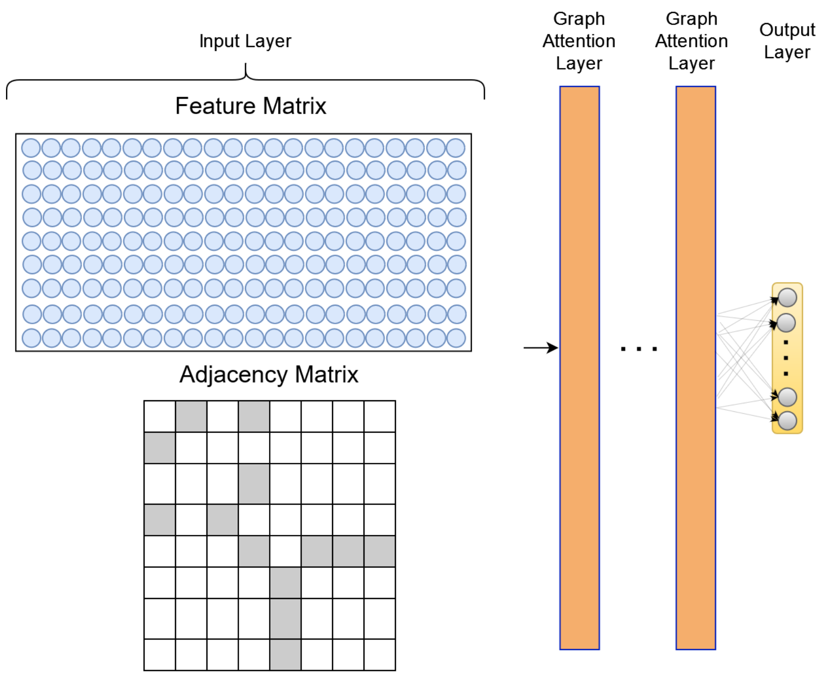


Figure 6.11 Architecture of the GAT model for text classification

The hidden layer of the GAT is computed as follows:

$$H^{i+1} = \sigma(A \cdot H^i \cdot W^i) \tag{20}$$

where Wi is the weight matrix in layer i, A is the adjacency matrix, H^i is the feature matrix of the hidden layer ($H^0 = X$), where X is the feature matrix extracted from the semantic graph or dependency tree, and σ refers to ReLU non-linear activation function.

In the proposed model, we apply a multi-layer GAT where the layer size is a hyperparameter that needs to be tuned in the graph.

Similar to the GCN model, the output layer differs depending on the problem. We feed the output of the node in the final layer into the softmax layer to learn the class of given text for binary and multi-class classification:

$$Z = softmax(H^l) \tag{21}$$

where H^l is the feature matrix of the final GAT layer.

For the multi-label classification problem, we used sigmoid function in the output layer that squeezes the results between 0 and 1, and we used 0.5 as a threshold to convert the probabilities into classes. The formula of the layer is in equation 22.

$$Z = sigmoid(H^1) \tag{22}$$

where H^l is the feature matrix of the final GAT layer.

6.3.2. Experiments & Results

6.3.2.1. Datasets

- **Irony Detection** For the irony detection problem, we focus on a low-resource language, Turkish, and conduct experiments on datasets released in Turkish [8, 9], which is new compared to other languages [187–190, 287]. There are two versions of the Turkish dataset. The first version [8] called Turkish-Irony-Dataset $(IronyTR_{220})$ is released in 2020 and the second one [9] IronyTR Dataset $(IronyTR_{600})$ is released in 2021. All data is collected from Twitter and other social media platforms. The details of the datasets are given in Table 6.4.

Dataset	Ironic	Non-Ironic	Total
Turkish-Irony-Dataset [8] $(IronyTR_{220})$	110	110	220
IronyTR Dataset [9] $(IronyTR_{600})$	300	300	600

Table 6.4 Turkish Datasets used in irony detection task

- **Multi-label Emotion Classification** We conduct experiments on 2 multi-label emotion classification datasets, i.e., SemEval-2018 Task-1C and GoEmotions. The details of the datasets are given below:
  – **SemEval-2018 Task-1C: Affect in Tweets** The dataset is developed for Task E-c: Detecting Emotions shared task [10], containing Tweets collected from 2016

to 2017. Although the dataset is provided in three languages (Arabic, Spanish, and English) for the emotion classification task, we used only English language data for this study.

- **GoEmotions:** A Dataset of Fine-Grained Emotions contains 58k samples from Reddit comments [6]. This dataset is annotated with the 28 emotions (27 emotion categories or neutral). We mapped the 28 labels to Ekman's [7] 6 categories. The mapping details are given in Table 6.5. We randomly split the dataset into the train (80%), development (10%), and test (10%) sets as defined in the paper [6].

Ekman	GoEmotions
anger	anger annoyance disapproval
disgust	disgust
fear	fear nervousness
joy	admiration amusement approval caring desire excitement gratitude joy optimism pride relief
sadness	disappointment embarrassment grief remorse sadness
neutral	neutral

Table 6.5 The mapping of the GoEmotions tagset [6] to the Ekman's tagset [7]

The statistics of the datasets are provided in Table 6.6, and Table 6.7 gives the percentage of texts annotated with given emotions in the datasets. The sum of the rows is more than 100% because a text in a multi-label dataset may have multiple emotions annotated. The percentage of emotions shows that anger, disgust, joy, optimism, and sadness received a higher percentage, while neutral, surprise and trust received a lower percentage in the SemEval-2018 Task-1C dataset [10]. With a similar percentage, joy and neutral received a higher percentage in the GoEmotions dataset [6].

Dataset	**Train Set**	**Dev Set**	**Test Set**	**Total**
SemEval-2018 Task-1C	6,838	886	3,259	10,983
GoEmotions	43,410	5,426	5,427	54,263

Table 6.6 Dataset statistics used in MLEC task

SemEval-2018 Task-1C		**Go Emotions**	
Emotion	**%**	**Emotion**	**%**
Anger	36.1	Anger	12.85
Anticipation	13.9	Disgust	1.83
Disgust	36.6	Fear	1.67
Fear	16.8	Joy	40.11
Joy	39.3	Sadness	7.52
Love	12.3	Surprise	12.36
Optimism	31.3	Neutral	32.76
Pessimism	11.6	-	-
Sadness	29.4	-	-
Surprise	5.2	-	-
Trust	5.0	-	-
Neutral	2.7	-	-

Table 6.7 Percentage of texts that were annotated with a given emotion in the MLEC datasets

- **Transfer Downstream Tasks** We also use the downstream tasks of the STS problem which are binary or multi-class classification tasks. We used the same datasets as described in Section 6.2..

6.3.2.2. Evaluation Metric We evaluated the models applied to binary and multi-class classification problems using the following 4 metrics: accuracy, precision, recall, and F1-score.

For multi-label classification, we reported Micro F1 and Macro F1 as well as multi-label accuracy, also called Jaccard index. This evaluation metric is calculated by dividing the size of the intersection of the predicted labels and the gold labels by the size of the union of the predicted and gold labels. This evaluation is performed for each text t in the test dataset and then averaged over all instances in dataset D.

$$Accuracy = \frac{1}{|D|} \sum_{t \in D} \frac{G_t \cap P_t}{G_t \cup P_t} \tag{23}$$

where G_t and P_s are the gold and predicted labels of the sentence s and D is the total number of sentences in the test set.

6.3.2.3. Hyperparameters and Implementation Details We use the PyTorch 3.7 package for the implementation of the GNNs. For the GNNs, we used the cross-entropy as the objective function in binary and multi-class classification problems and the BCE loss function as the objective function in multi-label classification. The Adam [108] is used as the optimizer in the models with an epsilon value of $1e - 8$ and the default max grad norm.

For the terminal nodes of the UCCA graphs and the nodes of the AMR and dependency trees, we use monolingual (BERT [91], RoBERTa [111], XLNet [112]) and multilingual pre-trained language models (M- BERT [91], XLM-R [113], XLM-R-large [114]) as embeddings in the feature matrix X. The hyperparameter values we found for all models in the problems can be found in Appendix 8.2..

We use the semantic parser described in Section 4.2.1. to extract the UCCA semantic graphs used in the proposed models. The model is trained using the SemEval 2019 shared task dataset [1]. We combine all training sets of datasets as train data of the model since previously conducted experiments with the model show that combining all languages as zero-shot experiments is more effective for low-resource languages [102]. We use the same hyperparameter set that was used in the experiments given in Section 4.2.1.. To extract AMR graphs and dependency trees of the datasets, we use the spaCy [279] and Stanza [281] libraries respectively.

6.3.2.4. Irony Detection Results We investigate the impact of the semantic framework on the success of the proposed models for low-resource language in irony detection. For this purpose, we conduct semantic and syntax-aware GNN model experiments for Turkish. The experiments are conducted under 10-fold cross-validation for the datasets as used in the cited papers [8, 9].

Table 6.8 and 6.9 give the results with state-of-the art results for the Turkish-Irony-Dataset [8] and the IronyTR Dataset [9], respectively.

Method	Accuracy	Precision	Recall	F1-Score
	Our proposed models			
UCCA-CNN	88.60	91.82	85.59	88.18
UCCA-GCN	**94.09**	**94.50**	93.64	94.06
UCCA-GAT	**94.09**	92.82	**95.45**	**94.17**
Dep-CNN	93.64	**93.64**	93.64	**93.64**
Dep-GCN	**94.12**	92.73	**94.55**	92.86
Dep-GAT	93.64	**93.64**	93.64	**93.64**
	State-of-the-art models			
Gauss NB [8]	57.50	40.00	20.00	26.67
Multinomial NB [8]	55.00	50.00	27.50	35.50
SVM [8]	60.50	66.00	30.00	41.25
BERT [8]	87.50	90.71	84.00	87.23
LSTM [8]	50.00	50.00	**100.00**	66.67

Table 6.8 Irony detection results using the proposed models and state-of-the-art results for the Turkish-Irony dataset [8]

We obtain the best F1 score with the UCCA-GAT and Dep-GCN models within the semantic and syntax-aware models for the Turkish-Irony Dataset [8] and the IronyTR Dataset [9], respectively. The results of the semantic and syntax-aware models are very similar.

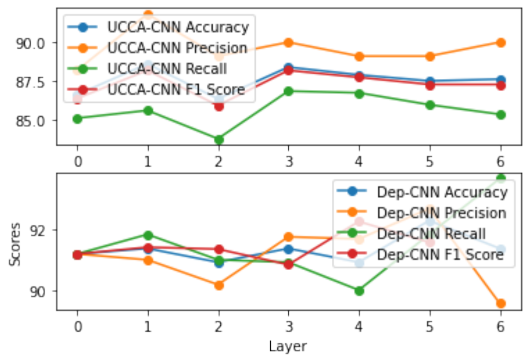
We also analyze the models deeply in terms of the impact of layers and embeddings.

- **Impact of Layers** Here, we analyze the impact of the number of layers of the proposed models on the performance (F1-Score) of the models. We perform the experiments with BERT-base embeddings. We vary the number of layers from 1 to 7 and report the results in Figure 6.12, 6.13 and 6.14 for semantic/syntax-aware CNN, GCN and GAT models, respectively.

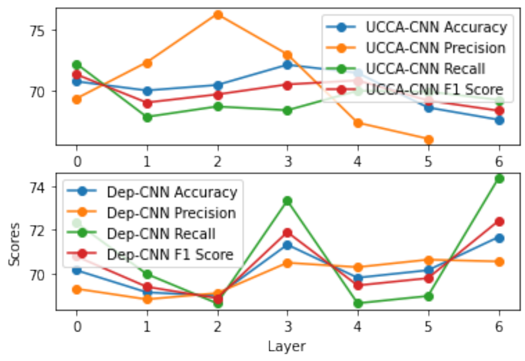
Method	Accuracy	Precision	Recall	F1-Score
	Our proposed models			
UCCA-CNN	70.75	69.33	72.22	71.33
UCCA-GCN	69.17	65.58	**80.67**	**72.35**
UCCA-GAT	**71.50**	**71.29**	72.00	71.64
Dep-CNN	71.67	70.57	**74.33**	72.40
Dep-GCN	73.00	72.70	73.67	**73.18**
Dep-GAT	**73.17**	**73.88**	71.67	72.76
	State-of-the-art models			
Basic [9]	55.33	58.83	68.44	63.27
Polarity [9]	55.83	58.23	74.35	65.31
Graph [9]	58.17	61.40	67.90	64.49
Pol-Gra [9]	55.67	58.11	76.12	65.91
LSTM [9]	51.33	55.09	52.73	53.88
LSTM+ [9]	50.50	49.88	64.57	56.28
Bi-LSTM [9]	50.16	51.44	62.07	56.26
Bi-LSTM+ [9]	51.66	52.61	56.74	54.60
CNN-LSTM [9]	50.33	50.33	45.73	47.92
CNN-LSTM+ [9]	50.33	46.73	45.59	46.15
BERT [9]	69.00	71.34	65.75	68.43

Table 6.9 Irony detection results using the proposed models and state-of-the-art results for IronyTR dataset [9]

- **CNN:** The Figure 6.12 shows that we achieve better performance in the 2nd and 6th layers of the UCCA-CNN and Dep-CNN architectures for the small dataset (Turkish-Irony-Dataset) and the 1st and 7th layers for the large dataset (IronyTR Dataset), respectively, than the others overall. We can conclude that while semantic-aware models learn with shallow models, syntax-aware models need deeper models for this problem.
- **GCN:** As shown in the Figure 6.13, we achieve better performance with the 2nd and 4th layers of UCCA-GCN and Dep-GCN architectures for the small dataset and the 5th and 3rd layers for the large dataset (IronyTR Dataset) than with the other layers overall. Semantic-aware model with one layer performs similarly for the small dataset. However, for the large dataset, the semantic-aware model with one layer performs worse, indicating that the shallow graph network structure is not able to learn ironic features well. When the number of layers is greater than 2,

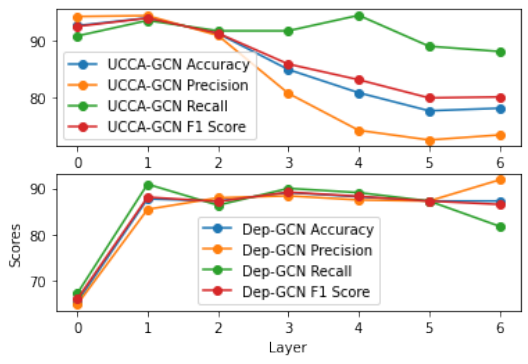


(a) Turkish-Irony-Dataset

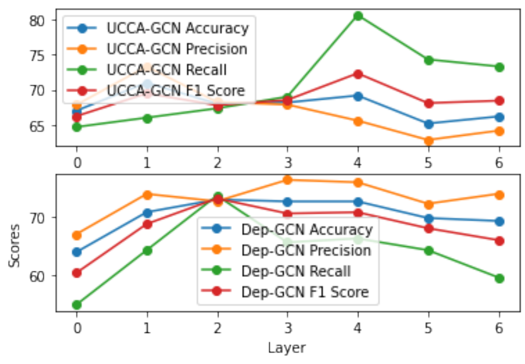


(b) IronyTR Dataset

Figure 6.12 Scores of semantic/syntax-aware CNN models



(a) Turkish-Irony-Dataset

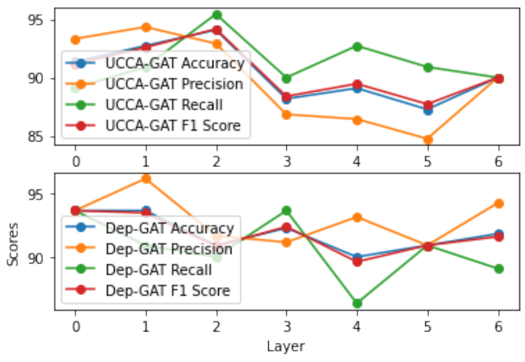


(b) IronyTR Dataset

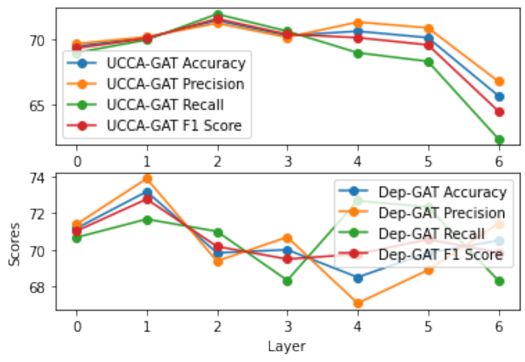
Figure 6.13 Scores of semantic/syntax-aware GCN models

the performance tends to decrease for small datasets. This indicates that further increasing the number of layers beyond 2 degrades the performance of the model, possibly due to the sharp increase in parameters. For the syntax-aware model, it can be seen that the deeper model tends to perform better on small and large datasets.

– **GAT:** We give the results in Figure 6.14. It can be seen that the 3rd layer of the UCCA-GAT architecture performs better overall than others for the two datasets. After the 3rd layer, there is a dramatic drop in results for the small dataset. This shows that deeper semantic-aware models lose the ironic features needed for the problem. However, for the large data set, the results do not decrease dramatically for a while. After the 6th layer, there is a dramatic drop in results. The Dep-GAT



(a) Turkish-Irony-Dataset



(b) IronyTR Dataset

Figure 6.14 Scores of semantic/syntax-aware GAT models

architecture gets better results with 1st and 2nd layers for the TurkishIrony and the IronyTR datasets. The shallow syntax-aware models perform better on this problem. While there is an increase after the 4th layer, it tends to decrease dramatically after the 7th layer for the larger dataset.

- **Embeddings** Following the success of the pre-trained language models, we used monolingual and multilingual embeddings in the experiments. As monolingual embeddings, we use the BERT-base pre-trained model (`bert-base-turkish-cased`). As multilingual embeddings, we use the multilingual version of the BERT-base model (M-BERT) [91], XLM-R [113], which is a multilingual RoBERTa model, and the large version of XLM (XLM-R-large) [114].

  The results of the models with embeddings are given in Table 6.10 and 6.11 for semantic and syntax-aware models respectively. While we obtain better results with semantic-aware models in monolingual experiments, the syntax-aware model gets similar results with the multilingual RoBERTa (XLM-R) and its large version in the small dataset.

6.3.2.5. Multi-Label Emotion Classification Results Tables 6.12 and 6.13 represent the multi-label accuracy, Micro F1, and Macro F1 results obtained by applying the proposed semantic and syntax-aware GAT (we used only the GAT model since we get the highest

	Turkish Irony Dataset [8]			IronyTR Dataset [9]		
Embeddings	UCCA-CNN	UCCA-GCN	UCCA-GAT	UCCA-CNN	UCCA-GCN	UCCA-GAT
Monolingual Embeddings						
BERT-base	**88.18**	**94.06**	**94.17**	**71.33**	**72.35**	**71.64**
Multilingual Embeddings						
M-BERT	88.00	91.20	85.45	69.01	68.48	68.00
XLM-R	87.89	90.55	87.73	67.02	67.45	68.16
XLM-R-large	87.32	90.32	86.15	70.33	68.68	69.95

Table 6.10 F1 scores of irony detection datasets with different embeddings

	Turkish Irony Dataset [8]			IronyTR Dataset [9]		
Embeddings	Dep-CNN	Dep-GCN	Dep-GAT	Dep-CNN	Dep-GCN	Dep-GAT
Monolingual Embeddings						
BERT-base	92.24	89.19	**93.64**	**72.40**	**73.18**	**72.76**
Multilingual Embeddings						
M-BERT	88.29	90.91	88.70	70.63	69.55	71.10
XLM-R	91.48	**92.86**	89.81	70.70	68.85	68.13
XLM-R-large	**93.64**	92.31	80.98	72.04	72.04	67.09

Table 6.11 F1 scores obtained with monolingual and multilingual embeddings

accuracy in irony detection) model on the SemEval-2018 Task-1C [10] and GoEmotions [6] datasets, respectively.

On SemEval-2018 Task-1C [10] dataset, overall, the best results are obtained with the semantic-aware UCCA-GAT model (accuracy = 61.2). Our proposed models, UCCA-GAT and Dep-GAT, outperformed the top three state-of-the-art approaches on the SemEval-2018 Task-1C dataset.

On GoEmotions [6] dataset, overall, the semantic-aware UCCA-GAT model performed best (accuracy = 71.2). Our proposed models, UCCA-GAT and Dep-GAT, outperformed the top three state-of-the-art studies on the GoEmotions dataset.

The results show that semantic and syntax-aware models are best suited for the MLEC problem for two different types of texts (tweets and Reddit comments).

To understand the performance of the models for each emotion, we computed the precision, recall, and macro F1 scores of the best performing model (UCCA-GAT) for the

Method	Accuracy	Micro F1	Macro F1
Our proposed models			
UCCA-GAT	**61.2**	66.1	**60.0**
Dep-GAT	59.7	63.5	57.8
Top three state-of-the-art models			
NTUA-SLP [201]	58.8	**70.1**	52.8
TCS Research [211]	58.2	69.3	53.0
PlusEmo2Vec [288]	57.6	69.2	49.7

Table 6.12 Emotion classification results obtained from SemEval-2018 Task-1C dataset [10]

Method	Accuracy	Micro F1	Macro F1
Our proposed models			
UCCA-GAT	**71.2**	**75.4**	63.9
Dep-GAT	68.7	74.7	61.1
Top three state-of-the-art models			
BERT [6]	-	-	**64.0**
RoBERTa [215]	65.9	69.1	61.8
Dim-RoBERTa [215]	65.7	68.6	61.0

Table 6.13 Emotion classification results obtained from the GoEmotions dataset [6]

SemEval-2018 Task-1C and GoEmotions datasets. The results are shown in Table 6.14 and 6.15 for the SemEval-2018 Task-1C and GoEmotions datasets, respectively.

	UCCA-GAT			**Dep-GAT**		
Emotion	Precision	Recall	F1	Precision	Recall	F1
Anger	**85.1**	**81.8**	**83.4**	75.6	72.7	74.1
Anticipation	**35.1**	**63.8**	**45.3**	26.9	52.0	35.5
Disgust	73.6	**89.0**	80.6	**74.8**	88.0	**80.8**
Fear	**32.3**	**76.3**	**45.4**	28.6	72.1	41.0
Joy	77.6	**91.2**	**83.9**	**82.5**	65.3	72.9
Love	**51.2**	**75.7**	**61.1**	50.4	73.3	59.7
Optimism	67.8	**85.2**	**75.5**	**70.4**	73.5	71.9
Pessimism	**81.3**	60.4	**69.3**	58.8	**76.0**	66.3
Sadness	**73.4**	**77.1**	**75.2**	64.3	71.4	67.7
Surprise	**52.9**	15.2	23.6	23.1	**52.9**	**32.1**
Trust	**54.2**	09.4	16.0	21.7	**54.2**	**31.0**

Table 6.14 Scores for each emotion in SemEval-2018 Task-1C dataset [10]

The results show that both semantic and syntax-aware models performed better on emotions “anger”, “disgust”, “joy”, “pessimism” and “sadness” on SemEval-2018 Task-1C dataset and

	UCCA-GAT			Dep-GAT		
Emotion	Precision	Recall	F1	Precision	Recall	F1
Anger	85.8	**84.0**	**84.9**	**90.7**	68.9	78.3
Disgust	**32.9**	**81.3**	**46.8**	27.6	78.9	40.9
Fear	**15.2**	**73.5**	**25.2**	11.3	56.1	18.9
Joy	**88.9**	76.2	82.0	84.9	**86.0**	**85.4**
Sadness	32.0	**68.0**	43.5	**41.1**	55.4	**47.2**
Surprise	**75.0**	88.6	**81.2**	64.2	**95.3**	76.8
Neutral	79.7	**87.9**	**83.6**	**82.8**	78.3	80.5

Table 6.15 Scores for each emotion in GoEmotions dataset [6]

emotions “anger”, “disgust”, “joy”, “optimism” in GoEmotions dataset. One possible reason for this could be the percentage of instances of these particular emotions in the datasets.

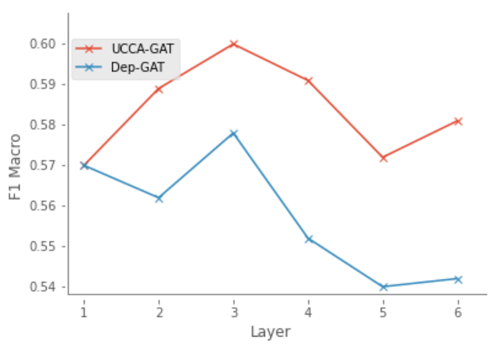
We also analyse the models deeply in terms of the impact of layers and embeddings.

- **Impact of Layers** To further analyze the layers’ impact on the proposed model, we varied the number of layers from 1 to 6 and performed experiments using BERT multilingual embeddings. Figure 6.15 illustrated the obtained results for the SemEval-2018 Task-1C and GoEmotions datasets.

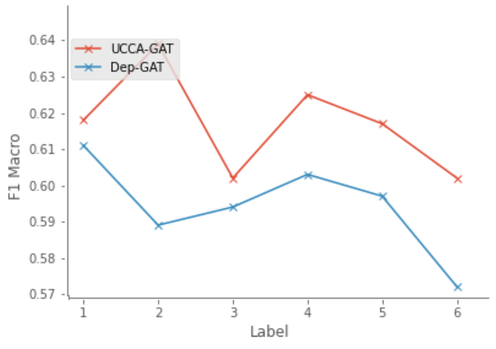
  As shown in Figure 6.15(a), we achieved the highest results on the 3rd layer of the UCCA-GAT and Dep-GAT models for the SemEval-2018 Task-1C dataset. After the 3rd layer, there is a dramatic drop in the F1 scores of the proposed models. This shows that deeper models lose the semantic features needed for the task. Figure 6.15(b) displayed the results of the GoEmotions dataset, where the highest performance can be seen on the 2nd layer of UCCA-GAT and 1st layer of Dep-GAT models. These results are similar to those of the SemEval-2018 Task-1C dataset. However, the dramatic drop can be seen in F1 scores of both proposed models (UCCA-GAT and Dep-GAT) after the 4th layer.

  The results obtained for two datasets show that the semantic and syntax-aware models (UCCA-GAT and Dep-GAT) do not need deeper layers for the MLEC problem.

  The results of both datasets show that the semantic and syntax-aware models (UCCA-GAT and Dep-GAT) do not require deeper layers for the MLEC problem.



(a) SemEval-2018 Task-1C Dataset



(b) GoEmotions dataset

Figure 6.15 Macro F1 scores obtained by proposed models with different number of layers on SemEval-2018 Task-1C [10] and GoEmotions [6] datasets

However, some limitations of this study are the dependency on external parser models, the difference between the domains used to train the parser models and the MLEC problem, and finally the emotions and punctuation marks in the MLEC datasets.

- **Embeddings** The effect of the embeddings on the models' success is huge [289]. Therefore, in our experiments, we tried monolingual and multilingual pre-trained embeddings to understand the behavior of semantic and syntax-aware models. We used BERT [91] (`bert-base-cased`), RoBERTa [111] (`roberta-base`), and XLNet [112] (`xlnet-base-cased`) monolingual embeddings with base variants consisting of 768 hidden dimensions, while we used multilingual versions of BERT (M-BERT) [91] and RoBERTa (XLM-R) [113].

  The results obtained by monolingual and multilingual pre-trained embeddings are given in Table 6.16. The results show that multilingual embeddings are more effective for the both proposed UCCA-GAT and Dep-GAT models.

6.3.2.6. Transfer Downstream Tasks Results Table 6.17 presents the accuracy results of the semantic and syntax-aware GAT models (we used only the GAT model because we get the best results in detecting irony with the GAT model) on 7 STS transfer downstream tasks defined in SentEval [217] with the state-of-the-art results.

	SemEval-2018 Task-1C dataset		**GoEmotions Dataset**	
PLM	UCCA-GAT	Dep-GAT	UCCA-GAT	Dep-GAT
Monolingual Embeddings				
BERT	58.0	56.3	61.5	58.9
RoBERTa	**59.2**	56.9	**62.7**	59.5
XLNet	56.5	54.0	59.2	57.0
Multilingual Embeddings				
M-BERT	58.5	57.8	63.5	61.1
XLM-R	**60.0**	56.2	**63.9**	60.2

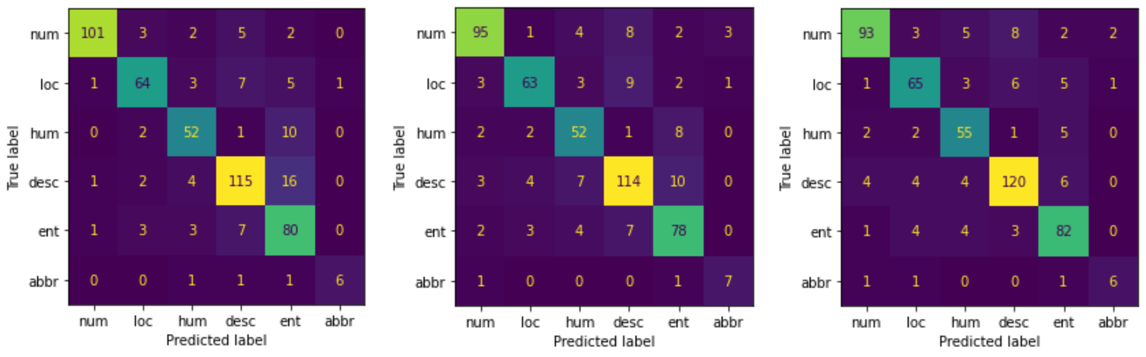
Table 6.16 F1 Scores obtained with monolingual and multilingual embeddings

The results show that the performance of the GAT model is far behind the state-of-the-art results. The main reason is that the models achieve state-of-the-art results, learn sentence embeddings, and then apply the learned sentence embeddings to transfer downstream tasks [3, 4]. Our proposed SiamseRvNN model (see Section 6.2.) also uses the same procedure. Therefore, we only compare the performance of the semantic and syntax-aware GAT (UCCA-GAT, AMR-GAT and Dep-GAT) models with each other for 7 transfer downstream tasks.

	Our proposed models							
	MR	**CR**	**SUBJ**	**MPQA**	**SST-2**	**TREC**	**MRPC**	**Avg.**
UCCA-GAT	82.04	83.37	90.38	87.29	89.35	81.92	73.50	83.98
AMR-GAT	81.55	81.11	88.98	83.94	85.83	79.65	72.87	83.42
Dep-GAT	80.66	81.62	89.10	85.76	88.03	81.06	75.25	83.07
State-of-the-art								
Siamese-RvNN	84.25	89.54	**94.68**	89.10	**91.25**	88.42	74.93	87.45
BERT-CLS embedding ♡	78.68	84.85	94.21	88.23	84.13	91.40	71.13	84.66
BiLSTM ♢	81.1	86.3	92.4	90.2	-	-	-	-
Universal Sentence Encoder ♣	80.09	85.19	93.98	86.70	86.38	93.2	70.14	85.10
SimCSE-BERT$_{base}$ ♠	83.64	89.43	94.39	89.86	88.96	**89.60**	**76.00**	87.41
SBERT-NLI-large ♡	**84.88**	**90.07**	94.52	**90.33**	90.66	87.4	75.94	**87.69**

Table 6.17 Results of transfer downstream tasks (measured as accuracy) using proposed models and state-of-the-art results (♣ results from [2]; ♠ results from [3]; ♡ results from [4]; ♢ results from [5])

The results show that the UCCA-GAT model is better than the AMR-GAT model, which is also a semantic-aware model. The analysis of the adjacency matrices extracted from the semantic AMR parsers shows that the relations such as about, like, of, etc. are defined as concepts and used as edge labels. Since our models use the nodes without edge labels, the



(a) UCCA-GAT (b) AMR-GAT (c) Dep-GAT

Figure 6.16 Confusion matrices of the semantic and syntax-aware GAT models on TREC dataset [11]

model misses the concepts that are important for the problem. This also leads to a sparse adjacency matrix compared to other adjacency matrices extracted from UCCA graphs and dependency trees.

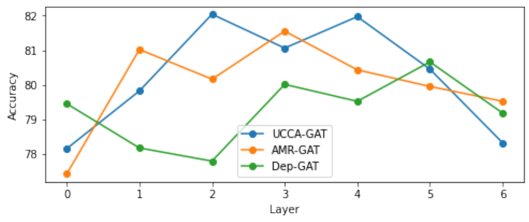
To understand the impact of the methods, we give the class-wise results of the semantic (UCCA/AMR) and syntax-aware (dependency) GAT models for the TREC dataset [11] (multi-class classification problem) in Table 6.18.

	UCCA-GAT			**AMR-GAT**			**Dep-GAT**		
Class	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
num	0.97	0.89	**0.93**	0.90	0.84	0.87	0.91	0.82	0.87
loc	0.86	0.79	**0.83**	0.86	0.78	0.82	0.82	0.80	0.81
hum	0.80	0.80	0.80	0.74	0.80	0.77	0.77	0.85	**0.81**
desc	0.85	0.83	0.84	0.82	0.83	0.82	0.87	0.87	**0.87**
enty	0.70	0.85	0.77	0.77	0.83	0.80	0.81	0.87	**0.84**
abbr	0.86	0.67	**0.75**	0.64	0.78	0.70	0.67	0.67	0.67

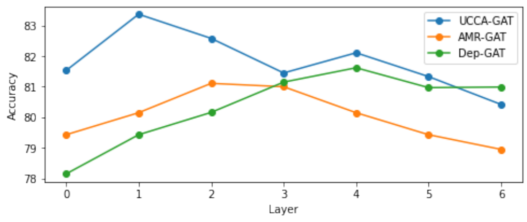
Table 6.18 Class-wise results of TREC dataset [11]

Figure 6.16 shows the matrices of the semantic and syntax-aware GAT models on the TREC dataset[11]. The results show that the UCCA-GAT model predicts the class `num` better than other models. In addition, the Dep- GAT model is better at predicting the class `desc`.

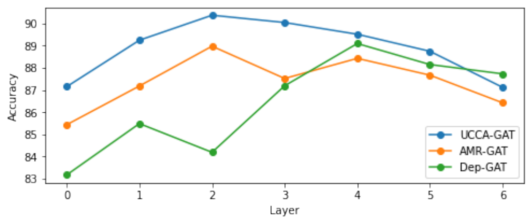
We also analyze the models deeply in terms of the impact of layers and embeddings.



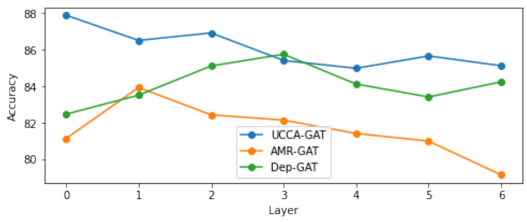
(a) MR Dataset



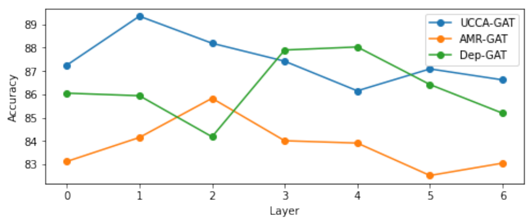
(b) CR Dataset



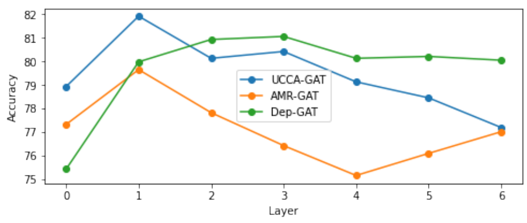
(c) SUBJ Dataset



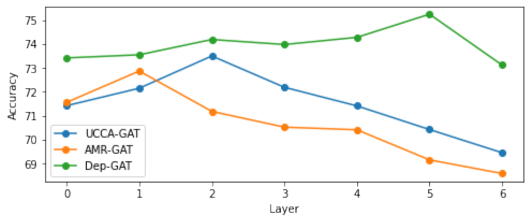
(d) MPQA Dataset



(e) SST-2 Dataset



(f) TREC Dataset



(g) MRPC Dataset

Figure 6.17 Scores of semantic/syntax-aware GAT models

- **Impact of Layers** Here, we analyze the impact of the number of layers of the proposed models (UCCA-GAT, AMR-GAT, Dep-GAT) on the performance (accuracy) of the models. We perform the experiments with embeddings in which we obtained the best results. We vary the number of layers from 1 to 7 and report the results in Figure 6.17 for all datasets with the UCCA-GAT, AMR-GAT, and Dep-GAT models. The results show that the syntax-aware model learns in deeper layers and semantic-aware models (UCCA-GAT and AMR-GAT) tend to learn in shallow layers or in middle layers.

- **Embeddings** We also present an analysis of pre-trained language models used in the extraction of feature matrices from UCCA, AMR, and dependency representations. We used BERT [91] (`bert-base-cased`), RoBERTa [111] (`roberta-base`), and XLNet [112] (`xlnet-base-cased`) monolingual embeddings with base variants, which include 768 hidden dimensions, whereas we used multilingual versions of BERT (M- BERT) [91], and RoBERTa (XLM-R) [113] and its large version (XLM-R-large) [113].

  The results obtained by monolingual and multilingual pre-trained embeddings are given in Table 6.19, 6.20 and 6.21 for UCCA-GAT, AMR-GAT, and Dep-GAT respectively. The results show that multilingual embeddings are more effective for both proposed semantic and syntax-aware models. For the monolingual embeddings, the results of RoBERTa pre-trained word embeddings are higher than the others (BERT, XLNet).

PLM	MR	CR	SUBJ	MPQA	SST-2	TREC	MRPC
Monolingual Embeddings							
BERT	78.33	79.15	87.80	82.78	85.01	80.40	69.68
RoBERTa	80.16	79.89	89.11	**87.29**	**89.35**	79.11	72.52
XLNet	74.62	75.99	83.15	77.46	80.56	76.82	67.71
Multilingual Embeddings							
M-BERT	79.27	81.94	88.15	83.11	83.14	81.00	72.35
XLM-R	**82.04**	82.23	89.48	84.76	85.01	**81.92**	72.93
XLM-R-large	78.78	**83.37**	**90.38**	85.82	87.59	81.42	**73.50**

Table 6.19 Accuracy results obtained with monolingual and multilingual embeddings in UCCA-GAT model

6.4. Question Answering

Question Answering (QA) is the task of producing generated answers to questions asked by a human in natural language. We propose semantic-aware BERT integrated semantic information as an attention mechanism to BERT for the question-answering problem.

PLM	MR	CR	SUBJ	MPQA	SST-2	TREC	MRPC
Monolingual Embeddings							
BERT	77.68	**81.11**	83.98	83.11	82.48	75.61	70.43
RoBERTa	**81.55**	79.44	85.44	83.44	**85.83**	**79.65**	70.78
XLNet	72.64	72.12	82.56	78.15	79.68	71.95	68.87
Multilingual Embeddings							
M-BERT	78.77	79.71	87.45	82.17	83.91	76.42	71.19
XLM-R	79.49	79.28	**88.98**	83.56	84.46	78.20	**72.87**
XLM-R-large	80.10	80.08	87.95	**83.94**	85.01	78.62	72.35

Table 6.20 Accuracy results obtained with monolingual and multilingual embeddings in AMR-GAT model

PLM	MR	CR	SUBJ	MPQA	SST-2	TREC	MRPC
Monolingual Embeddings							
BERT	77.30	79.50	86.43	82.99	83.64	78.80	70.78
RoBERTa	78.95	**80.11**	89.10	83.14	**88.03**	79.62	71.59
XLNet	72.45	74.40	82.47	78.04	81.38	75.27	69.51
Multilingual Embeddings							
M-BERT	79.39	79.55	84.69	82.64	84.51	79.89	73.51
XLM-R	80.19	81.62	87.59	83.84	85.78	**81.06**	74.09
XLM-R-large	**80.66**	81.14	**88.11**	**85.76**	86.49	79.49	**75.25**

Table 6.21 Accuracy results obtained with monolingual and multilingual embeddings in Dep-GAT model

6.4.1. Proposed Method

With the inspiration of syntax-aware BERT [12], we use graph-based semantic representation, UCCA, for local attention and integrate it with standard Transformer attention.

6.4.1.1. Semantic-aware Local Attention Local attention involves limiting each token to attend to a subset of the other tokens in the input. Recent studies employing local attention use fixed or dynamic windows to derive the important local regions [249, 290, 291]. Here, we apply semantic-aware local attention, where the semantic structure is derived from the UCCA graph-based semantic representation.

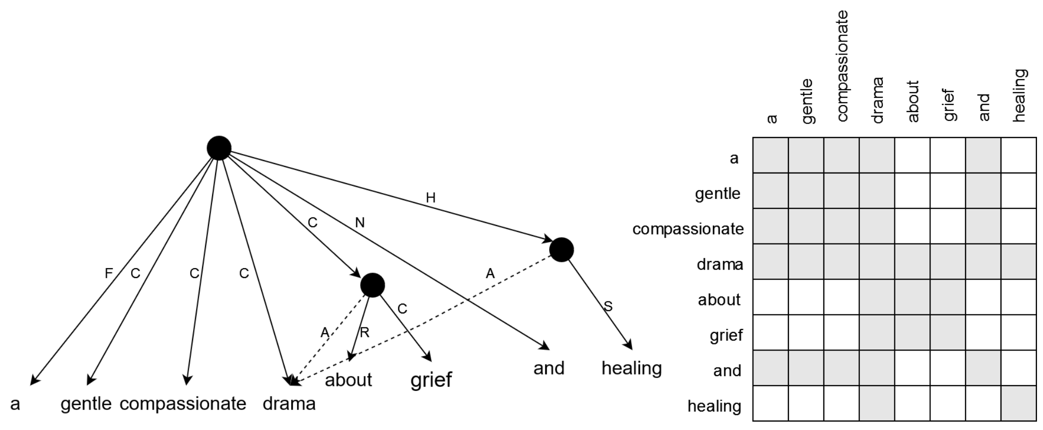


Figure 6.18 The adjacency matrix extracted from UCCA representation of the sample *"a gentle compassionate drama about grief and healing"* taken from MR dataset [13]. The gray color in the matrix represents the value of 1 and the white color to the value of 0.

The UCCA representation of a sentence is represented as a graph $G = (V, E, X)$, where V is the set of nodes, E is the set of labeled edges representing the UCCA relations, and X is the set of non-terminal tokens of the sentence. Since the UCCA graph has terminal nodes that are tokens of the sentence and non-terminal nodes compromising several tokens that are jointly considered as a single entity. We consider adjacent words to be connected to the same non-terminal node. An example of an adjacency matrix extracted from the UCCA representation of a sentence *"a gentle compassionate drama about grief and healing"* is given in Figure 6.18.

In the representation, each token x_i is mapped to a node vi and the distance from node v_i to v_j is denoted by $dis(v_i, v_j)$. The distance from token x_i to x_j is calculated as follows:

$$D(i,j) = min\, dis(v_k, v_j),\ k \in [i-1, i+2] \tag{24}$$

Semantic-aware local attention with given query $\mathbf{Q}$ and key $\mathbf{K}$ is computed as follows:

$$\begin{aligned}\mathbf{M}^{loc}_{ij} &= \begin{cases} 0 & D(i,j) \leq m \\ -\infty & \text{prevent from attending} \end{cases} \\ \mathbf{S}^{loc} &= softmax(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}} + \mathbf{M}^{loc})\end{aligned} \tag{25}$$

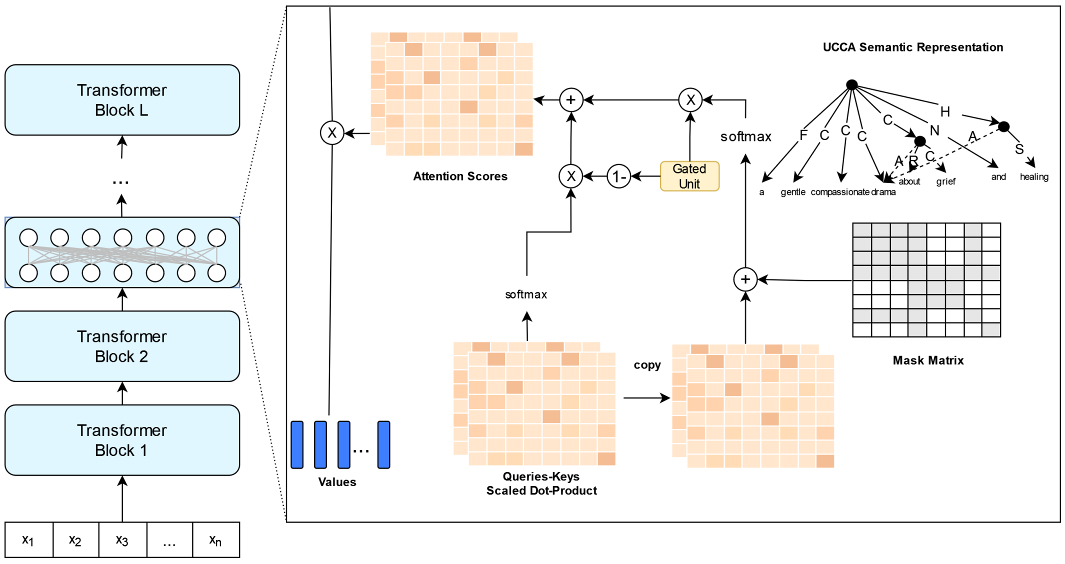


Figure 6.19 Overview of the semantic-aware BERT model

where m is a threshold to restrict the distance $D(i, j)$, and d is the dimension of keys.

Finally, we compute the attention score by combining the local attention with the standard global attention as given follow:

$$\begin{aligned} g_i &= \sigma(\mathbf{W}_g h_i + b_g) \\ \hat{\mathbf{A}}_i &= (g_i S_i^{loc} + (1 - g_i) S_i^{glb}) \mathbf{V} \end{aligned} \tag{26}$$

where g_i is the gate value calculated for each token x_i, h_i is the hidden vector of token x_i from the previous layer, W_g is a learnable linear transformation, b_g is the bias. $\hat{\mathbf{A}}_i$ is calculated as a weighted average over values $\mathbf{V}$ with scores of local and global attention.

The architecture of the semantic-aware BERT model is given in Figure 6.19

6.4.1.2. Semantic-BERT for Question Answering We applied semantic-BERT for QA, which is commonly solved by BERT [91]. For QA, we use the semantic-aware BERT with two parameters, the input question, and the passage containing the answer of the question, similar to BERT. The architecture of the semantic-BERT for QA is given in Figure 6.20.

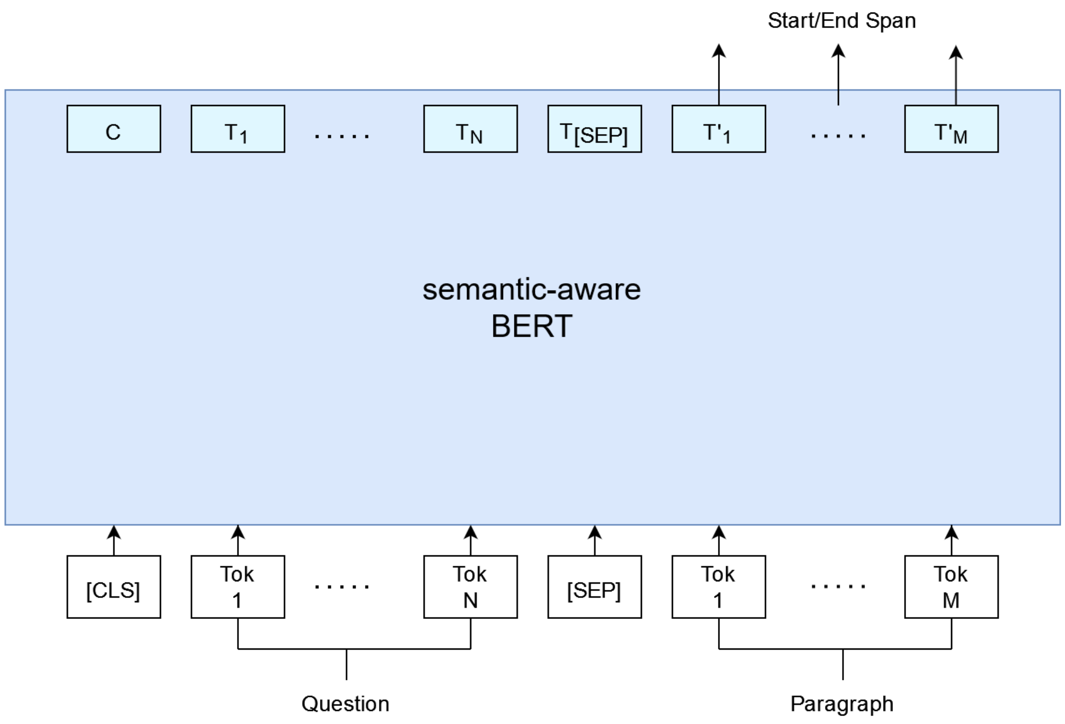


Figure 6.20 Architecture of semantic-BERT model for QA

Dataset	Title	Context	Question
Train	756	2,400	14,221
Dev	133	541	2,520

Table 6.22 Statistical details of the QA dataset

6.4.2. Experiments & Results

6.4.2.1. Dataset We evaluated the Semantic-BERT model on the Turkish Question Answering dataset (TQuAD)[27]. It is a QA dataset that contains question-answer pairs created through crowd-sourcing, using Wikipedia as a base. The details of the dataset are given in Table 6.22.

6.4.2.2. Evaluation Metric We use Exact Match (EM) and F1 score, the standard evaluation metrics of QA. EM represents the percentage of questions where the predicted

[27]`https://huggingface.co/datasets/husnu/tquad2`

answer exactly matches one of the correct answers for the question, and F1 score measures the average overlap between the predicted and correct answers for the questions.

6.4.2.3. Hyperparameters and Implementation Details We use the semantic parser described in Section 4.2.1. to extract UCCA semantic representations of the datasets. The few-shot experiment setting of the Turkish UCCA dataset is used to train the model. The details of the experiments and hyperparameters are given in Section 4.3..

The inputs of the BERT model are tokenized using the WordPiece tokenizer, where words can be split into several sub-words. Since the tokens of the UCCA semantic representation are words, we apply the same masking value to the sub-words split by the WordPiece tokenizer in the semantic-aware local attention. We use `bert-base-turkish-cased` and `bert-base-multilingual-cased` BERT for the experiments. The maximum length is set to 512. We use the Adam optimizer [108] for training. The number of epochs is 5 and the batch size is 16 for the problem. We used a small learning rate ($1e-5$), similar to previous studies [12].

6.4.2.4. Results We conducted experiments on the TQuAD dataset and reported the performance of the proposed method in Table 6.23 with the results of the BERT model.

	Turkish		**Multilingual**	
	EM	F1	EM	F1
BERT [91]	36.3	56.9	41.7	62.6
Syntax-aware BERT [12]	40.9	59.4	48.3	66.8
Semantic-aware BERT	**42.4**	**61.6**	**50.1**	**68.7**

Table 6.23 Task Performance on QA

According to the results of fine-tuning the TQuAD dataset in Table 6.23, the multilingual BERT achieves better results than Turkish for QA. This is due to the larger size of train set of the multilingual model. Moreover, the semantic information integrated into local attention increases EM and F1 results for Turkish and multilingual models for BERT compared to the syntax-aware BERT. The semantic information in Semantic-BERT increases the EM results by up to 1.5 and 1.8 and F1 results by up to 2.2 and 1.9 compared to syntax-aware BERT for Turkish and multilingual models. Since the UCCA representation can consist of multiple sentences, usually one paragraph or multiple paragraphs [17], the UCCA representation covers the context of the question better than the dependency tree used in syntax-aware BERT.

Question:
Elway 38 yaşındayken hangi Super Bowl'u kazanmıştır?
Answer: Super Bowl XXXIII
semantic-aware BERT: Super Bowl XXXIII
syntax-aware BERT [12]: Super Bowl

Question:
Hürjet projesinin hangi bir diğer proje ile arasında bir sanayi geçişi sağlanacağı belirtilmiştir?
Answer: Milli Muharip Uçak Projesi
semantic-aware BERT: Milli Muharip Uçak
syntax-aware BERT [12]: Uçak Projesi

Question:
Piri Reis'in Osmanlı İmparatorluğu'nun Akdeniz Hakimiyetini sağlamak için yaptığı seferlerin başlangıç noktası neresidir?
Answer: Gelibolu
semantic-aware BERT: Eğriboz Bahriye Azapları
syntax-aware BERT [12]: Mısır

Question:
Piri Reis Gelibolu'da dünya haritasını kaç yılında hazırlamıştır?
Answer: 1513
semantic-aware BERT: (1513)
syntax-aware BERT [12]: (1554)

Table 6.24 Sample results taken from proposed model semantic-aware BERT and syntax-aware BERT [12]

Since the multilingual BERT performs better in the TQuAD dataset, the results of the semantic-BERT are not surprising. In addition, we have used semantic parser trained in few-shot experiments (see Section 4.3.) in which we used English, German, and Turkish train set to extract the UCCA representation of the TQuAD dataset. This may also have an important impact on the semantic-BERT results.

When analysing the errors of semantic-aware and syntax-aware BERT, it is clear that semantic-aware BERT obtains more accurate answers from larger contexts for the given question than syntax-aware BERT. In particular, the semantic-aware BERT is better at predicting answers that are phrases. Some of the errors of the semantic-aware BERT are due to the incorrect UCCA representation of context and question caused by the semantic parsing model. The errors of the semantic parsing model degrade the performance of the semantic-aware BERT. Some errors of semantic-aware and syntax-aware BERT with correct answers are given in Table 6.24.

6.5. Discussion

The answer to the questions we asked at the beginning of this chapter.

1. Can semantic-aware models using UCCA representation perform relatively well on NLP tasks compared to other current SOTA methods that do not utilise any structures semantic information?

   To evaluate the semantic-aware models, we proposed semantic-aware models for various NLP tasks that require semantic information. In the STS task, we applied Siamese-RvNN using the UCCA semantic representation to generate semantically informed sentence embeddings. The results of the STS task showed that the semantically-informed model outperformed the SOTA methods [2–4], indicating that the UCCA semantic annotation helps to learn better sentence embeddings. We also used the semantic-aware sentence embeddings for transfer downstream tasks defined in the SentEval toolkit [217]. The model achieved the best performance at 2 of 7 tasks, which have a larger train set within the downstream tasks.

   For the text classification problem, we proposed semantic aware GNN models. We applied the proposed methods to a wide range of datasets (binary, multi-class and multi-label classification) for Turkish and English. The irony detection problem for Turkish has comparatively small datasets (Turkish-Irony [8] and IronyTR [9]) and the problem is not well studied. The semantic-aware models (UCCA-CNN, UCCA-GCN and UCCA-GAT) outperformed the SOTA methods [8, 9]. For MLEC problem, we only applied the UCCA-GAT model since we obtained the best results by UCCA-GAT in the irony detection problem. We used 2 different datasets for the problem, namely SemEval-2018 Task-1C [10] and GoEmotions [6]. We obtained the highest Accuracy results for the datasets and outperformed the SOTA methods for SemEval-2018 Task-1C [201, 211, 288] and Go-Emotion [6, 215]. The MLEC datasets are noisy compared to other datasets. The SemEval-2018 Task-1C dataset consists of tweets that contain hashtags, emotions (labels in text), and punctuation marks, while the GoEmotions dataset consists of Reddit comments that contain fewer emotion words and no hashtags compared to SemEval-2018. This may be one of the reasons for the worse results of the SemEval dataset compared to the GoEmotions dataset. Another reason may be the length of the text in the datasets (Average no. of words: SemEval-2018 Task-1C = 16.06, GoEmotions = 12.84) since the experimental results of the parsers show that the models perform better on shorter texts than on longer ones [102]. The last reason could be the number of emotion classes in the datasets (SemEval-2018 Task-1C: 11, and GoEmotions: 7), which could lead to the low performance of the proposed model on the SemEval-2018 Task-1C dataset. Finally, we applied the semantic-aware model (UCCA-GAT) to downstream tasks of the SentEval toolkit [217]. We couldn't outperform the SOTA methods [2–4]. One of the methods is our Siamese-RvNN model. The main reason for the low results is that the SOTA methods were mainly proposed for STS and then the learned sentence embeddings were applied to downstream tasks. By applying the semantic-aware models on different problems for Turkish and English with different size of datasets, we get an insight into the limitations of the proposed model.

For QA, we proposed semantic-BERT in which an adjacency matrix extracted from the UCCA representation is integrated into the local attention mechanism of BERT [91]. We obtain better results using semantic-BERT it also proves the importance of semantic information in the models.

2. Are there differences in terms of the performance between semantic-aware and syntax-aware models on NLP tasks?

To answer this question, we applied the semantic and syntax-aware models to the text classification problem. We applied the dependency trees as syntactic information in the GNN models as syntax-aware models. Initially, we applied the dependency trees and the UCCA representation in the irony detection problem, which is a binary classification problem in Turkish. The syntax-aware model performed better in the larger dataset (IronyTR [9]) compared to the semantic-aware model. However, the size of the irony detection datasets is very small. Therefore, we applied the semantic-aware and syntax-aware models for the MLEC, which is a multi-label classification problem, using only the GAT model, since we obtained the highest accuracy results on the irony detection problem with this model. For the MLEC with larger datasets in English, we achieved the highest accuracy scores using the semantic-aware model with 61.2 accuracy (1.5 gain over the syntax-aware model) for the SemEval-2018 Task -1C dataset [10] and 71.2 accuracy (2.5 gain over the syntax-aware model) for the GoEmotions dataset [6]. Finally, we applied these models to downstream tasks of the SentEval toolkit [217], which are binary and multi-class classification problems. The semantic-aware models perform better on these problems, with the exception of the MRPC dataset, which is similar to the irony detection and MLEC results.

The experimental results show that the syntax-aware model (Dep-GAT) learns in deeper layers and semantic-aware models (UCCA-GAT and AMR-GAT) tend to learn in shallow layers or in the middle layers. Since syntactic features are encoded in the shallow layers and semantic features are encoded in the deeper layers of the pre-trained language models [292, 293], we obtained better results for the syntax-aware model with deeper layers and for the semantic-aware models (UCCA-GAT and AMR-GAT) with shallow layers.

3. Do semantic-aware models with different graph-based semantic representations achieve different results on NLP problems?

   We applied UCCA, which is the main topic of this thesis, and the AMR as another semantic representation. We chose the AMR representation because both semantic representations are based on DAG and the AMR representation is a well-studied representation in the literature. We compared these representations in downstream tasks of the SentEval toolkit [217]. The results of the semantic-aware model with UCCA representation were comparatively better than those of the model with AMR representation. We applied a similar method to extract adjacency and feature matrices from the AMR representation, in which the adjacency matrix was comparatively sparse because we only used the nodes of the graphs. However, the edges of the AMR are also very informative due to the core roles extracted from the PropBank.

4. Do the semantic-aware models using graph-based semantic representations achieve different results in NLP problems? Is the NLP problem important for the performance of the semantic-aware models?

   NLP problems mainly suffer from capturing the semantic of the text in an abstract way. The recent studies on various NLP problems such as summarization [34, 242] and NMT [25, 246] proposed methods with semantic representations to address this problem. We also applied the semantic-aware models to 3 different problems: STS, text classification, and QA, to answer this question. For 3 problems, the results prove that incorporating semantic information into the models improves the performance of the models. The important part of the proposed models for the problems is that the models are very capable of integrating semantic information. In the STS task, we applied Siamese-RvNN, where RvNN uses a recursive structure in the tree hierarchy (see Section 6.2.). For text classification, we used GNN models that take adjacency and feature matrix extracted from graphs as input(see Section 6.3.). Finally, we used semantic-BERT, where we integrated semantic information into the local matrix as the adjacency matrix for QA (see Section 6.4.).

6.6. Summary

In this chapter, we focus on semantic-aware models for NLP problems as an extrinsic evaluation of the UCCA representation. We introduced models for STS, text classification, and QA. We showed that semantic-aware models using the UCCA semantic representation outperform SOTA methods for problems, proving that semantic information improves the performance of the models for the problem.

We also investigated syntactic information and AMR semantic representation as semantic information in text classification. The results of the semantic-aware model with AMR and the syntax-aware model with dependency tree are comparatively lower than those of the semantic-aware model with UCCA representation.

Finally, we applied the semantic-aware BERT model to the QA problem, but it can be applied to any NLP problem using BERT.

Our contributions in this chapter are as follows:

- We introduced a neural network architecture Siamese-RvNN for evaluating the semantic textual similarity between two sentences.
- Our proposed Siamese-RvNN model outperformed other approaches in the STS task. We also obtained competitive results in transfer learning tasks by using semantically informed sentence embeddings in downstream NLP tasks such as sentiment analysis.
- We proposed semantic and syntax-aware GNN models for text classification. To the best of our knowledge, this is the first study to integrate the UCCA framework into GNNs for text classification and to compare the effects of semantic and syntactic representations for this problem.
- We compared semantic and syntax-aware models for text classification in a wide range of problems (binary, multi-class, and multi-label) and languages (Turkish and English) in detail. The experimental results showed that the semantic-aware model outperformed the syntax-aware models. Moreover, the semantic-aware model outperformed the state-of-the-art performance on the datasets.
- We proposed a semantic-aware BERT model for QA. This is the first attempt to integrate the UCCA representation into the BERT model in the attention mechanism.

Experimental results show that semantic information improves the performance of the BERT model for QA.

7. CONCLUSION

7.1. Contributions

In this thesis, we focused on UCCA representation from different perspectives, especially for the Turkish language. We showed that UCCA graph-based semantic representation is powerful for language understanding. We have discussed the properties of UCCA and devised different neural parsing algorithms for UCCA representation. Furthermore, we proposed a parser model (self-attentive) that allowed us to annotate a dataset in a semi-automatic pipeline process for Turkish which is a low-resource language. The proposed Turkish dataset is an important contribution even though the dataset is comparatively smaller than other UCCA datasets annotated in other languages which are the *English Wikipedia* [39] and *English 20K Leagues Under The Sea* [133]. Also, the provided guideline, which covers the grammatical rules of Turkish that are different from English, helps researchers to extend the dataset. The method can also be helpful for researchers to annotate a dataset in different languages. We also evaluated the UCCA representation as a source of semantic information by using it for various NLP problems with a comparison between semantic and syntactic representations.

In Chapter 4., we proposed three different parsing models for UCCA representation. Different approaches are applied to understand the impact of transition-based and graph-based parsers for UCCA representation. Our graph-based model yielded comparatively better improvement in F1 scores for UCCA representation compared to other studies in the literature. The incremental parser obtained better results than the TUPA parser [126], a transition-based parser which is the first parser introduced for UCCA representation. The experiments performed with these two models also provided a detailed analysis of the impact of the size of the dataset. The last presented model is an unsupervised parsing model. To the best of our knowledge, this is the first time that an unsupervised parsing model is applied to the UCCA representation, with an in-depth analysis of the model for semantic and syntactic parsing problems, namely semantic, dependency, and constituency parsing.

In Chapter 5., we proposed a semi-automatic pipeline for dataset annotation. We took advantage of the UCCA representation, which is a cross-linguistic annotation scheme, and used an external semantic parser proposed in Chapter 4. to obtain an initial annotation for the raw sentences. After analysing the parsed sentences as the output of the parser model, we determined the discrepancies between the Turkish and English rules defined in the English UCCA guideline. Accordingly, we defined rules for Turkish grammar.

In Chapter 6., we presented semantic-aware models for various NLP problems, namely STS, text classification, and QA. For the STS problem, we used the Siamese model with RvNN called Siamese-RvNN, which uses the UCCA representation for the STS datasets. For text classification, we used GNN models with semantic information and analysed the impact of UCCA representation on the problem by using AMR representation as another semantic representation and dependency trees as syntactic representation. Finally, we applied the semantic-BERT by integrating the UCCA representation with local attention aggregated with attention of BERT for QA.

7.2. Challenges

The UCCA representation is a comparatively new representation in the literature and studies are limited compared to other graph-based semantic representations such as AMR, EDS. Moreover, the properties of the UCCA representation (reentrancy, discontinuous unit, non-terminal nodes) differ from other syntactic representations in the parsing problem.

The dataset annotation in this thesis is a major contribution to the literature. Annotation of datasets is usually critical, as even the slightest error can lead to bigger errors. Moreover, the annotation process is time-consuming, expensive, and requires expertise. Since UCCA is a cross-linguistic annotation scheme, annotation may be easier than other representations. However, there are still discrepancies between Turkish and English, for which there is a detailed annotation guideline.

Another challenge we encountered is building semantic and syntax-aware models using the trained parsing models. Since the problem of dependency parsing is older than semantic parsers, many of the proposed models have very effective results [294–296]. Moreover, the dataset size of AMR representations and dependency trees is larger than that of UCCA representations. The final challenge is the accumulated error due to the use of external parsers for the semantic and syntax-aware models since the models use the parsers to obtain semantic and syntactic information from the datasets.

7.3. Further Analysis

Future work on UCCA representation includes improving the current parsing models using semantic representations with similar properties such as DAG, discontinuity, etc., to learn representation in-framework settings to leverage the models.

For the Turkish UCCA dataset, an annotated dataset at the morpheme level will be targeted in the future because Turkish is an agglutinative language and new words are formed by adding morphemes to roots. Moreover, morphemes in Turkish are also meaningful units and should be incorporated in the semantic representation. Moreover, the foundational layer of UCCA can be easily extended to include morphemes as defined by Abend and Rappoport [17].

Exploring the contribution of UCCA in various NLP problems is also a very important research topic, as the results presented in Chapter 6. demonstrate the importance of semantic information in NLP problems.

REFERENCES

[1] Daniel Hershcovich, Zohar Aizenbud, Leshem Choshen, Elior Sulem, Ari Rappoport, and Omri Abend. SemEval-2019 Task 1: Cross-lingual Semantic Parsing with UCCA. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1–10. **2019**.

[2] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. Universal Sentence Encoder. *CoRR*, abs/1803.11175, **2018**.

[3] Tianyu Gao, Xingcheng Yao, and Danqi Chen. SimCSE: Simple Contrastive Learning of Sentence Embeddings. *CoRR*, abs/2104.08821, **2021**.

[4] Nils Reimers, Iryna Gurevych, Nils Reimers, Iryna Gurevych, Nandan Thakur, Nils Reimers, Johannes Daxenberger, and Iryna Gurevych. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, **2019**.

[5] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. Supervised Learning of Universal Sentence Representations from Natural Language Inference Data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680. **2017**.

[6] Dorottya Demszky, Dana Movshovitz-Attias, Jeongwoo Ko, Alan S. Cowen, Gaurav Nemade, and Sujith Ravi. GoEmotions: A Dataset of Fine-Grained Emotions. *CoRR*, abs/2005.00547, **2020**.

[7] Paul Ekman and Harriet Oster. Facial Expressions of Emotion. *Annual review of psychology*, 30(1):527–554, **1979**.

[8] Cenk Cidecio, Yeşim Cemek, Aslı Umay Öztürk, RECEP FIRAT ÇEKİNEL, and PINAR KARAGÖZ. Türkçe Resmi Olmayan Metinlerde ironi Tespiti için Sinirsel Yöntemlerin incelenmesi. **2020**.

[9] Asli Umay Ozturk, Yesim Cemek, and Pinar Karagoz. IronyTR: Irony Detection in Turkish Informal Texts. *International Journal of Intelligent Information Technologies (IJIIT)*, 17(4):1–18, **2021**.

[10] Saif M. Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. SemEval-2018 Task 1: Affect in Tweets. In *Proceedings of International Workshop on Semantic Evaluation (SemEval-2018)*. New Orleans, LA, USA, **2018**.

[11] Ellen M Voorhees and Dawn M Tice. Building a question answering test collection. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 200–207. **2000**.

[12] Zhongli Li, Qingyu Zhou, Chao Li, Ke Xu, and Yunbo Cao. Improving BERT with Syntax-aware Local Attention. *CoRR*, abs/2012.15150, **2020**.

[13] Bo Pang and Lillian Lee. Seeing Stars: Exploiting Class Relationships for Sentiment Categorization with Respect to Rating Scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 115–124. **2005**.

[14] Omri Abend and Ari Rappoport. The state of the art in semantic representation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 77–89. **2017**.

[15] Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. Abstract meaning representation for sembanking. In *Proceedings*

of the 7th linguistic annotation workshop and interoperability with discourse, pages 178–186. **2013**.

[16] Stephan Oepen and Jan Tore Lønning. Discriminant-Based MRS Banking. In *LREC*, pages 1250–1255. **2006**.

[17] Omri Abend and Ari Rappoport. UCCA: A Semantics-based Grammatical Annotation Scheme. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013)–Long Papers*, pages 1–12. **2013**.

[18] Aaron Steven White, Drew Reisinger, Keisuke Sakaguchi, Tim Vieira, Sheng Zhang, Rachel Rudinger, Kyle Rawlins, and Benjamin Van Durme. Universal decompositional semantics on universal dependencies. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1713–1723. **2016**.

[19] Fei Liu, Jeffrey Flanigan, Sam Thomson, Norman Sadeh, and Noah A Smith. Toward Abstractive Summarization Using Semantic Representations. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1077–1086. **2015**.

[20] Kexin Liao, Logan Lebanoff, and Fei Liu. Abstract Meaning Representation for Multi-Document Summarization. pages 1178–1190, **2018**.

[21] Xuan Zhang, Huizhou Zhao, KeXin Zhang, and Yiyang Zhang. SEMA: Text Simplification Evaluation through Semantic Alignment. In *Proceedings of the 6th Workshop on Natural Language Processing Techniques for Educational Applications*, pages 121–128. **2020**.

[22] Fuad Issa, Marco Damonte, Shay B Cohen, Xiaohui Yan, and Yi Chang. Abstract meaning representation for paraphrase detection. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for*

Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pages 442–452. **2018**.

[23] Linfeng Song, Daniel Gildea, Yue Zhang, Zhiguo Wang, and Jinsong Su. Semantic neural machine translation using AMR. *Transactions of the Association for Computational Linguistics*, 7:19–31, **2019**.

[24] Elior Sulem, Omri Abend, and Ari Rappoport. Semantic structural decomposition for neural machine translation. In *Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics*, pages 50–57. **2020**.

[25] Long HB Nguyen, Viet H Pham, and Dien Dinh. Improving neural machine translation with AMR semantic graphs. *Mathematical Problems in Engineering*, 2021, **2021**.

[26] Weiwen Xu, Huihui Zhang, Deng Cai, and Wai Lam. Dynamic Semantic Graph Construction and Reasoning for Explainable Multi-hop Science Question Answering. *CoRR*, abs/2105.11776, **2021**.

[27] Pavan Kapanipathi, Ibrahim Abdelaziz, Srinivas Ravishankar, Salim Roukos, Alexander Gray, Ramón Fernandez Astudillo, Maria Chang, Cristina Cornelio, Saswati Dana, Achille Fokoue-Nkoutche, et al. Leveraging Abstract Meaning Representation for Knowledge Base Question Answering. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3884–3894. **2021**.

[28] Tahira Naseem, Srinivas Ravishankar, Nandana Mihindukulasooriya, Ibrahim Abdelaziz, Young-Suk Lee, Pavan Kapanipathi, Salim Roukos, Alfio Gliozzo, and Alexander Gray. A semantics-aware transformer model of relation linking for knowledge base question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 256–262. **2021**.

[29] Elior Sulem, Omri Abend, and Ari Rappoport. Simple and Effective Text Simplification Using Semantic and Neural Methods. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 162–173. **2018**.

[30] Ronald Langacker. *Cognitive Grammar: A Basic Introduction*. **2008**.

[31] Martha Palmer, Daniel Gildea, and Paul Kingsbury. The proposition bank: An annotated corpus of semantic roles. *Computational linguistics*, 31(1):71–106, **2005**.

[32] Elif Oral, Ali Acar, and Gülşen Eryiğit. Abstract Meaning Representation of Turkish. *Natural Language Engineering*, pages 1–30, **2022**.

[33] Shibhansh Dohare and Harish Karnick. Text Summarization using Abstract Meaning Representation. *CoRR*, abs/1706.01678, **2017**.

[34] Fei Liu, Jeffrey Flanigan, Sam Thomson, Norman M. Sadeh, and Noah A. Smith. Toward Abstractive Summarization Using Semantic Representations. *CoRR*, abs/1805.10399, **2018**.

[35] Pavan Kapanipathi, Ibrahim Abdelaziz, Srinivas Ravishankar, Salim Roukos, Alexander G. Gray, Ramón Fernandez Astudillo, Maria Chang, Cristina Cornelio, Saswati Dana, Achille Fokoue, Dinesh Garg, Alfio Gliozzo, Sairam Gurajada, Hima Karanam, Naweed Khan, Dinesh Khandelwal, Young-Suk Lee, Yunyao Li, Francois P. S. Luus, Ndivhuwo Makondo, Nandana Mihindukulasooriya, Tahira Naseem, Sumit Neelam, Lucian Popa, Revanth Gangi Reddy, Ryan Riegel, Gaetano Rossiello, Udit Sharma, G. P. Shrivatsa Bhargav, and Mo Yu. Question Answering over Knowledge Bases by Leveraging Semantic Parsing and Neuro-Symbolic Reasoning. *CoRR*, abs/2012.01707, **2020**.

[36] Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Silvie Cinková, Dan Flickinger, Jan Hajic, Angelina Ivanova, and Zdenka Uresova.

Towards comparability of linguistic graph banks for semantic parsing. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3991–3995. **2016**.

[37] Lasha Abzianidze, Johannes Bjerva, Kilian Evang, Hessel Haagsma, Rik van Noord, Pierre Ludmann, Duc-Duy Nguyen, and Johan Bos. The Parallel Meaning Bank: Towards a Multilingual Corpus of Translations Annotated with Compositional Meaning Representations. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 242–247. **2017**.

[38] Marco Kuhlmann and Stephan Oepen. Towards a catalogue of linguistic graph banks. *Computational Linguistics*, 42(4):819–827, **2016**.

[39] Omri Abend and Ari Rappoport. Universal Conceptual Cognitive Annotation (UCCA). In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 228–238. **2013**.

[40] Stephan Oepen, Omri Abend, Jan Hajic, Daniel Hershcovich, Marco Kuhlmann, Tim O'Gorman, Nianwen Xue, Jayeol Chun, Milan Straka, and Zdenka Urešová. MRP 2019: Cross-framework Meaning Representation Parsing. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 1–27. **2019**.

[41] Jakob Prange, Nathan Schneider, and Omri Abend. Made for Each Other: Broad-coverage Semantic Structures Meet Preposition Supersenses. *CoRR*, abs/1909.08796, **2019**.

[42] Adi Shalev, Jena D Hwang, Nathan Schneider, Vivek Srikumar, Omri Abend, and Ari Rappoport. Preparing SNACS for subjects and objects. In *Proceedings of the First International Workshop on Designing Meaning Representations*, pages 141–147. **2019**.

[43] Robert Malcolm Ward Dixon and RMW Dixon. *Basic linguistic theory volume 2: Grammatical Topics*, volume 2. Oxford University Press, **2010a**.

[44] Joakim Nivre. An efficient algorithm for projective dependency parsing. In *Proceedings of the eighth international conference on parsing technologies*, pages 149–160. **2003**.

[45] Joakim Nivre. Algorithms for deterministic incremental dependency parsing. *Computational Linguistics*, 34(4):513–553, **2008**.

[46] Daniel Hershcovich, Omri Abend, and Ari Rappoport. A Transition-Based Directed Acyclic Graph Parser for UCCA. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1127–1138. **2017**.

[47] Kenji Sagae and Alon Lavie. A classifier-based parser with linear run-time complexity. In *Proceedings of the Ninth International Workshop on Parsing Technology*, pages 125–132. **2005**.

[48] Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajic. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of human language technology conference and conference on empirical methods in natural language processing*, pages 523–530. **2005**.

[49] Frank Drewes, H-J Kreowski, and Annegret Habel. Hyperedge replacement graph grammars. In *Handbook Of Graph Grammars And Computing By Graph Transformation: Volume 1: Foundations*, pages 95–162. World Scientific, **1997**.

[50] Jonas Groschwitz, Meaghan Fowlie, Mark Johnson, and Alexander Koller. A constrained graph algebra for semantic parsing with AMRs. In *IWCS 2017-12th International Conference on Computational Semantics-Long papers*. **2017**.

[51] Paul Kingsbury and Martha Palmer. From Treebank to PropBank. In *LREC*, pages 1989–1993. Citeseer, **2002**.

[52] Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Dan Flickinger, Jan Hajič, Angelina Ivanova, and Yi Zhang. SemEval 2014 Task 8: Broad-Coverage Semantic Dependency Parsing. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 63–72. Association for Computational Linguistics, Dublin, Ireland, **2014**.

[53] Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Silvie Cinková, Dan Flickinger, Jan Hajic, and Zdenka Uresova. SemEval 2015 task 18: Broad-coverage semantic dependency parsing. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 915–926. **2015**.

[54] Geoffrey L. Lewis. Turkish grammar, **1967**.

[55] Robert B Lees. The Turkish copula. In *The Formal Analysis of Natural Languages*, pages 175–179. De Gruyter Mouton, **2018**.

[56] Jaklin Kornfilt. *Turkish and the Turkic Languages*. **2018**.

[57] Aslı Göksel and Celia Kerslake. *Turkish: A Comprehensive Grammar*. **2004**.

[58] Wolfgang Maier. Discontinuous incremental shift-reduce parsing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1202–1212. **2015**.

[59] Michael Collins and Brian Roark. Incremental parsing with the perceptron algorithm. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 111–118. **2004**.

[60] Yoav Goldberg and Michael Elhadad. Learning sparser perceptron models. In *Acl*. MIT Press Journals, **2011**.

[61] Daniel Hershcovich, Omri Abend, and Ari Rappoport. Multitask Parsing Across Semantic Representations. In *Proceedings of the 56th Annual Meeting of the*

Association for Computational Linguistics (Volume 1: Long Papers), pages 373–385. **2018**.

[62] Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajic, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis M. Tyers, and Daniel Zeman. Universal Dependencies v2: An Evergrowing Multilingual Treebank Collection. *CoRR*, abs/2004.10643, **2020**.

[63] Joakim Nivre, Marie-Catherine De Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, et al. Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1659–1666. **2016**.

[64] Stephan Oepen, Omri Abend, Lasha Abzianidze, Johan Bos, Jan Hajic, Daniel Hershcovich, Bin Li, Tim O'Gorman, Nianwen Xue, and Daniel Zeman. MRP 2020: The Second Shared Task on Cross-Framework and Cross-Lingual Meaning Representation Parsing. In *Proceedings of the CoNLL 2020 Shared Task: Cross-Framework Meaning Representation Parsing*, pages 1–22. **2020**.

[65] Tobias Pütz and Kevin Glocker. Tüpa at SemEval-2019 Task1:(Almost) feature-free Semantic Parsing. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 113–118. **2019**.

[66] Weimin Lyu, Sheng Huang, Abdul Rafae Khan, Shengqiang Zhang, Weiwei Sun, and Jia Xu. CUNY-PKU Parser at SemEval-2019 Task 1: Cross-Lingual Semantic Parsing with UCCA. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 92–96. **2019**.

[67] Longxu Dou, Yunlong Feng, Yuqiu Ji, Wanxiang Che, and Ting Liu. HIT-SCIR at MRP 2020: Transition-based parser and iterative inference parser. In *Proceedings of the CoNLL 2020 Shared Task: Cross-Framework Meaning Representation Parsing*, pages 65–72. **2020**.

[68] Ofir Arviv, Ruixiang Cui, and Daniel Hershcovich. HUJI-KU at MRP 2020: Two Transition-based Neural Parsers. In *Proceedings of the CoNLL 2020 Shared Task: Cross-Framework Meaning Representation Parsing*, pages 73–82. **2020**.

[69] Sunny Lai, Chun Hei Lo, Kwong Sak Leung, and Yee Leung. CUHK at MRP 2019: Transition-based parser with cross-framework variable-arity resolve action. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 104–113. **2019**.

[70] Hongxiao Bai and Hai Zhao. SJTU at MRP 2019: A transition-based multi-task parser for cross-framework meaning representation parsing. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 86–94. **2019**.

[71] Wanxiang Che, Longxu Dou, Yang Xu, Yuxuan Wang, Yijia Liu, and Ting Liu. HIT-SCIR at MRP 2019: A unified pipeline for meaning representation parsing via efficient training and effective encoding. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 76–85. **2019**.

[72] Ciprian Chelba, Tomás Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, and Phillipp Koehn. One Billion Word Benchmark for Measuring Progress in Statistical Language Modeling. *CoRR*, abs/1312.3005, **2013**.

[73] Wei Jiang, Zhenghua Li, Yu Zhang, and Min Zhang. HLT@SUDA at SemEval-2019 Task 1: UCCA Graph Parsing as Constituent Tree Parsing. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 11–15. **2019**.

[74] Jie Cao, Yi Zhang, Adel Youssef, and Vivek Srikumar. Amazon at MRP 2019: Parsing Meaning Representations with Lexical and Phrasal Anchoring. In

Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning, pages 138–148. **2019**.

[75] Kira Droganova, Andrey Kutuzov, Nikita Mediankin, and Daniel Zeman. ÚFAL-Oslo at MRP 2019: Garage Sale Semantic Parsing. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 158–165. **2019**.

[76] Yuta Koreeda, Gaku Morio, Terufumi Morishita, Hiroaki Ozaki, and Kohsuke Yanai. Hitachi at MRP 2019: Unified Encoder-to-Biaffine Network for Cross-Framework Meaning Representation Parsing. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 114–126. **2019**.

[77] Zuchao Li, Hai Zhao, Zhuosheng Zhang, Rui Wang, Masao Utiyama, and Eiichiro Sumita. SJTU-NICT at MRP 2019: Multi-Task Learning for End-to-End Uniform Semantic Graph Parsing. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 45–54. **2019**.

[78] Seung-Hoon Na, Jinwoon Min, Kwanghyeon Park, Jong-Hun Shin, and Young-Gil Kim. JBNU at MRP 2019: Multi-level Biaffine Attention for Semantic Dependency Parsing. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 95–103. **2019**.

[79] Xinyu Wang, Jingxian Huang, and Kewei Tu. Second-Order Semantic Dependency Parsing with End-to-End Neural Networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4609–4618. **2019**.

[80] Yue Zhang, Wei Jiang, Qingrong Xia, Junjie Cao, Rui Wang, Zhenghua Li, and Min Zhang. SUDA-Alibaba at MRP 2019: Graph-Based Models with BERT. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 149–157. **2019**.

[81] Hao Peng, Sam Thomson, and Noah A Smith. Deep Multitask Learning for Semantic Dependency Parsing. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2037–2048. **2017**.

[82] Jeffrey Flanigan, Sam Thomson, Jaime G Carbonell, Chris Dyer, and Noah A Smith. A discriminative graph-based parser for the abstract meaning representation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1426–1436. **2014**.

[83] Milan Straka and Jana Straková. Tokenizing, pos tagging, lemmatizing and parsing UD 2.0 with udpipe. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 88–99. **2017**.

[84] Mitchell Stern, Jacob Andreas, and Dan Klein. A Minimal Span-Based Neural Constituency Parser. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 818–827. **2017**.

[85] Lucia Donatelli, Meaghan Fowlie, Jonas Groschwitz, Alexander Koller, Matthias Lindemann, Mario Mina, and Pia Weißenhorn. Saarland at MRP 2019: Compositional parsing across all graphbanks. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 66–75. **2019**.

[86] Matthias Lindemann, Jonas Groschwitz, and Alexander Koller. Compositional Semantic Parsing Across Graphbanks. *CoRR*, abs/1906.11746, **2019**.

[87] Dian Yu and Kenji Sagae. UC Davis at Semeval-2019 Task 1: DAG semantic parsing with attention-based decoder. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 119–124. **2019**.

[88] Hiroaki Ozaki, Gaku Morio, Yuta Koreeda, Terufumi Morishita, and Toshinori Miyoshi. Hitachi at MRP 2020: Text-to-graph-notation transducer. In *Proceedings of the CoNLL 2020 Shared Task: Cross-Framework Meaning Representation Parsing*, pages 40–52. **2020**.

[89] Nikita Kitaev and Dan Klein. Constituency Parsing with a Self-Attentive Encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2676–2686. **2018**.

[90] James Cross and Liang Huang. Span-Based Constituency Parsing with a Structure-Label System and Provably Optimal Dynamic Oracles. *CoRR*, abs/1612.06475, **2016**.

[91] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186. **2019**.

[92] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008. **2017**.

[93] Nikita Kitaev and Dan Klein. Multilingual Constituency Parsing with Self-Attention and Pre-Training. *CoRR*, abs/1812.11760, **2018**.

[94] J-C Chappelier and Martin Rajman. A generalized CYK algorithm for parsing stochastic CFG. In *Proc. of 1st Workshop on Tabulation in Parsing and Deduction (TAPD'98)*, CONF, pages 133–137. **1998**.

[95] Kaiyu Yang and Jia Deng. Strongly Incremental Constituency Parsing with Graph Neural Networks. *Advances in Neural Information Processing Systems*, 33:21687–21698, **2020**.

[96] Yikang Shen, Zhouhan Lin, Chin-Wei Huang, and Aaron C. Courville. Neural Language Modeling by Jointly Learning Syntax and Lexicon. *CoRR*, abs/1711.02013, **2017**.

[97] Yikang Shen, Zhouhan Lin, Athul Paul Jacob, Alessandro Sordoni, Aaron Courville, and Yoshua Bengio. Straight to the Tree: Constituency Parsing with Neural Syntactic Distance. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1171–1180. Association for Computational Linguistics, Melbourne, Australia, **2018**.

[98] Taeuk Kim, Jihun Choi, Sang-goo Lee, and Daniel Edmiston. Are Pre-trained Language Models Aware of Phrases? Simple but Strong Baselines for Grammar Induction. In *International Conference of Learning Representations*. **2020**.

[99] Taeuk Kim and Sang-goo Lee. Multilingual Zero-shot Constituency Parsing. *CoRR*, abs/2004.13805, **2020**.

[100] Bowen Li, Taeuk Kim, Reinald Kim Amplayo, and Frank Keller. Heads-up! Unsupervised Constituency Parsing via Self-Attention Heads. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 409–424. Suzhou, China, **2020**.

[101] Jason M. Eisner. Three New Probabilistic Models for Dependency Parsing: An Exploration. In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*. **1996**.

[102] Necva Bölücü and Burcu Can. Self-Attentive Constituency Parsing for UCCA-based Semantic Parsing. *CoRR*, abs/2110.00621, **2021**.

[103] Nart B Atalay, Kemal Oflazer, and Bilge Say. The annotation process in the Turkish treebank. In *Proceedings of 4th International Workshop on Linguistically Interpreted Corpora (LINC-03) at EACL 2003*. **2003**.

[104] Kemal Oflazer, Bilge Say, Dilek Zeynep Hakkani-Tür, and Gökhan Tür. Building a Turkish treebank. In *Treebanks*, pages 261–277. Springer, **2003**.

[105] Mary Ann Marcinkiewicz. Building a Large Annotated Corpus of English: The Penn Treebank. *Using Large Corpora*, page 273, **1994**.

[106] Djamé Seddah, Reut Tsarfaty, Sandra Kübler, Marie Candito, Jinho D Choi, Richárd Farkas, Jennifer Foster, Iakes Goenaga, Koldo Gojenola Galletebeitia, Yoav Goldberg, et al. Overview of the SPMRL 2013 Shared Task: A Cross-Framework Evaluation of Parsing Morphologically Rich Languages. In *Proceedings of the Fourth Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 146–182. Association for Computational Linguistics, **2013**.

[107] Olcay Taner Yıldız, Ercan Solak, Şemsinur Çandır, Razieh Ehsani, and Onur Görgün. Constructing a Turkish Constituency Parse TreeBank. In *Information Sciences and Systems 2015*, pages 339–347. Springer, **2016**.

[108] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization, **2014**.

[109] Piotr Bojanowski, Édouard Grave, Armand Joulin, and Tomáš Mikolov. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics*, 5:135–146, **2017**.

[110] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language Models are Unsupervised Multitask Learners. *OpenAI blog*, 1(8):9, **2019**.

[111] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR*, abs/1907.11692, **2019**.

[112] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32, **2019**.

[113] Alexis Conneau and Guillaume Lample. Cross-lingual Language Model Pretraining. *Advances in neural information processing systems*, 32, **2019**.

[114] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised Cross-lingual Representation Learning at Scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451. Online, **2020**.

[115] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, **2017**.

[116] Yuxuan Wang, Wanxiang Che, Jiang Guo, Yijia Liu, and Ting Liu. Cross-Lingual BERT Transformation for Zero-Shot Dependency Parsing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5725–5731. **2019**.

[117] Ke M Tran and Arianna Bisazza. Zero-shot Dependency Parsing with Pre-trained Multilingual Sentence Representations. In *Proceedings of the 2nd*

Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019), pages 281–288. **2019**.

[118] Anne Lauscher, Vinit Ravishankar, Ivan Vulić, and Goran Glavaš. From Zero to Hero: On the Limitations of Zero-Shot Language Transfer with Multilingual Transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4483–4499. **2020**.

[119] Yue Zhang and Stephen Clark. Transition-based parsing of the Chinese treebank using a global discriminative model. In *Proceedings of the 11th International Conference on Parsing Technologies (IWPT'09)*, pages 162–171. **2009**.

[120] Ryan McDonald and Joakim Nivre. Characterizing the Errors of Data-Driven Dependency Parsing Models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 122–131. **2007**.

[121] Johannes Gontrum. Attention mechanisms for transition-based dependency parsing, **2019**.

[122] Xuezhe Ma and Fei Xia. Unsupervised Dependency Parsing with Transferring Distribution via Parallel Guidance and Entropy Regularization. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1337–1348. **2014**.

[123] Yikang Shen, Yi Tay, Che Zheng, Dara Bahri, Donald Metzler, and Aaron Courville. StructFormer: Joint Unsupervised Induction of Dependency and Constituency Structure from Masked Language Modeling. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7196–7209. Online, **2021**.

[124] Dan Klein and Christopher Manning. Corpus-Based Induction of Syntactic Structure: Models of Dependency and Constituency. In *Proceedings of the 42nd*

Annual Meeting of the Association for Computational Linguistics (ACL-04), pages 478–485. Barcelona, Spain, **2004**.

[125] Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. What Does BERT Look at? An Analysis of BERT's Attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286. Association for Computational Linguistics, Florence, Italy, **2019**.

[126] Daniel Hershcovich and Ofir Arviv. TUPA at MRP 2019: A multi-task baseline system. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 28–39. **2019**.

[127] Haşim Sak, Tunga Güngör, and Murat Saraçlar. Resources for Turkish morphological processing. *Language resources and evaluation*, 45(2):249–261, **2011**.

[128] Kemal Oflazer. Turkish and its challenges for language processing. *Language resources and evaluation*, 48(4):639–653, **2014**.

[129] Zahra Azin and Gülşen Eryiğit. Towards Turkish Abstract Meaning Representation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 43–47. **2019**.

[130] Umut Sulubacak, Gülşen Eryiğit, Tuğba Pamay, et al. IMST: A Revisited Turkish Dependency Treebank. In *Proceedings of TurCLing 2016, the 1st International Conference on Turkic Computational Linguistics*. **2016**.

[131] Gözde Gül Şahin and Eşref Adalı. Annotation of semantic roles for the Turkish Proposition Bank. *Language Resources and Evaluation*, 52(3):673–706, **2018**.

[132] Umut Sulubacak and Gülşen Eryiğit. Implementing universal dependency, morphology, and multiword expression annotation standards for Turkish

language processing. *Turkish Journal of Electrical Engineering & Computer Sciences*, 26(3):1662–1672, **2018**.

[133] Elior Sulem, Omri Abend, and Ari Rappoport. Conceptual annotations preserve structure across translations: A French-English case study. In *Proceedings of the 1st Workshop on Semantics-Driven Statistical Machine Translation (S2MT 2015)*, pages 11–22. **2015**.

[134] Daniel Hershcovich, Omri Abend, and Ari Rappoport. Content Differences in Syntactic and Semantic Representation. In *Proc. of NAACL-HLT*, pages 478–488. **2019**.

[135] Omri Abend, Nathan Schneider, Dotan Dvir, Jakob Prange, and Ari Rappoport. UCCA's Foundational Layer: Annotation Guidelines v2.1. *CoRR*, abs/2012.15810, **2020**.

[136] Elvan Göçmen, Onur Sehitoglu, and Cem Bozsahin. An Outline of Turkish Syntax, **1995**.

[137] Deniz Zeyrek and Bonnie Webber. A Discourse Resource for Turkish: Annotating Discourse Connectives in the METU Corpus. In *Proceedings of the 6th workshop on Asian language resources*. **2008**.

[138] Jacob Cohen. A Coefficient of Agreement for Nominal Scales. *Educational and psychological measurement*, 20(1):37–46, **1960**.

[139] Yuhua Li, David McLean, Zuhair A Bandar, James D O'shea, and Keeley Crockett. Sentence Similarity Based on Semantic Nets and Corpus Statistics. *IEEE transactions on knowledge and data engineering*, 18(8):1138–1150, **2006**.

[140] Rada Mihalcea, Courtney Corley, Carlo Strapparava, et al. Corpus-based and Knowledge-based Measures of Text Semantic Similarity. In *AAAI*, volume 6, pages 775–780. **2006**.

[141] Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. SemEval-2012 task 6: A pilot on semantic textual similarity. In ** SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 385–393. **2012**.

[142] Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. SEM 2013 shared task: Semantic textual similarity. In *Second joint conference on lexical and computational semantics (* SEM), volume 1: proceedings of the Main conference and the shared task: semantic textual similarity*, pages 32–43. **2013**.

[143] Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. SemEval-2014 Task 10: Multilingual Semantic Textual Similarity. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*, pages 81–91. **2014**.

[144] Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Inigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, et al. SemEval-2015 task 2: Semantic textual similarity, English, Spanish and pilot on interpretability. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pages 252–263. **2015**.

[145] Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez Agirre, Rada Mihalcea, German Rigau Claramunt, and Janyce Wiebe. SemEval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In *SemEval-2016. 10th International Workshop on Semantic Evaluation; 2016 Jun 16-17; San Diego, CA. Stroudsburg (PA): ACL; 2016. p. 497-511.* ACL (Association for Computational Linguistics), **2016**.

[146] Frane Šarić, Goran Glavaš, Mladen Karan, Jan Šnajder, and Bojana Dalbelo Bašić. Takelab: Systems for measuring semantic text similarity. In ** SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 441–448. **2012**.

[147] Hua He, Kevin Gimpel, and Jimmy Lin. Multi-perspective sentence similarity modeling with convolutional neural networks. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1576–1586. **2015**.

[148] Naveed Afzal, Yanshan Wang, and Hongfang Liu. MayoNLP at Semeval-2016 Task 1: Semantic textual similarity based on lexical semantic net and deep learning semantic model. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 674–679. **2016**.

[149] Hua He and Jimmy Lin. Pairwise word interaction modeling with deep neural networks for semantic similarity measurement. In *Proceedings of the 2016 conference of the north American chapter of the Association for Computational Linguistics: human language technologies*, pages 937–948. **2016**.

[150] Yang Shao. HCTI at SemEval-2017 Task 1: Use convolutional neural network to evaluate semantic textual similarity. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 130–133. **2017**.

[151] Goutam Majumder, Partha Pakray, Alexander Gelbukh, and David Pinto. Semantic textual similarity methods, tools, and applications: A survey. *Computación y Sistemas*, 20(4):647–665, **2016**.

[152] Michael Sussna. Word Sense Disambiguation for Free-text Indexing Using a Massive Semantic Network. In *Proceedings of the second international conference on Information and knowledge management*, pages 67–74. **1993**.

[153] Jay J Jiang and David W Conrath. Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy. In *Proceedings of the 10th Research on Computational Linguistics International Conference*, pages 19–33. **1997**.

[154] Daniel Ramage, Anna N Rafferty, and Christopher D Manning. Random Walks for Text Semantic Similarity. In *Proceedings of the 2009 workshop on graph-based methods for natural language processing (TextGraphs-4)*, pages 23–31. **2009**.

[155] David Sánchez, Montserrat Batet, David Isern, and Aida Valls. Ontology-based semantic similarity: A new feature-based approach. *Expert systems with applications*, 39(9):7718–7728, **2012**.

[156] Yoan Gutiérrez, Sonia Vázquez, and Andrés Montoyo. A semantic framework for textual data enrichment. *Expert Systems with Applications*, 57:248–269, **2016**.

[157] Rie Kubota Ando. Latent Semantic Space: Iterative Scaling Improves Precision of Inter-document Similarity Measurement. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 216–223. **2000**.

[158] Evgeniy Gabrilovich, Shaul Markovitch, et al. Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis. In *IJcAI*, volume 7, pages 1606–1611. **2007**.

[159] Sergio Jimenez, Claudia Becerra, and Alexander Gelbukh. Soft cardinality: A parameterized similarity function for text comparison. In ** SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 449–453. **2012**.

[160] Md Arafat Sultan, Steven Bethard, and Tamara Sumner. DLS@ CU at Semeval-2016 Task 1: Supervised models of sentence similarity. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 650–655. **2016**.

[161] Jorge Martinez-Gil and Jose M Chaves-Gonzalez. A novel method based on symbolic regression for interpretable semantic similarity measurement. *Expert Systems with Applications*, 160:113663, **2020**.

[162] Xing Wu, Chaochen Gao, Liangjun Zang, Jizhong Han, Zhongyuan Wang, and Songlin Hu. ESimCSE: Enhanced Sample Building Method for Contrastive Learning of Unsupervised Sentence Embedding. *CoRR*, abs/2109.04380, **2021**.

[163] Junlei Zhang and Zhenzhong Lan. S-SimCSE: Sampled Sub-networks for Contrastive Learning of Sentence Embedding. *CoRR*, abs/2111.11750, **2021**.

[164] Kun Zhou, Beichen Zhang, Wayne Xin Zhao, and Ji-Rong Wen. Debiased Contrastive Learning of Unsupervised Sentence Representations, **2022**.

[165] Yiren Jian, Chongyang Gao, and Soroush Vosoughi. Non-Linguistic Supervision for Contrastive Learning of Sentence Embeddings, **2022**.

[166] Yung-Sung Chuang, Rumen Dangovski, Hongyin Luo, Yang Zhang, Shiyu Chang, Marin Soljačić, Shang-Wen Li, Wen-tau Yih, Yoon Kim, and James Glass. DiffCSE: Difference-based Contrastive Learning for Sentence Embeddings, **2022**.

[167] Yoon Kim. Convolutional Neural Networks for Sentence Classification. *CoRR*, abs/1408.5882, **2014**.

[168] Nguyen Huy Tien, Nguyen Minh Le, Yamasaki Tomohiro, and Izuha Tatsuya. Sentence Modeling via Multiple Word Embeddings and Multi-level Comparison for Semantic Textual Similarity. *Information Processing & Management*, 56(6):102090, **2019**.

[169] Hua He and Jimmy Lin. Pairwise Word Interaction Modeling with Deep Neural Networks for Semantic Similarity Measurement. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 937–948. Association for Computational Linguistics, San Diego, California, **2016**.

[170] Kai Sheng Tai, Richard Socher, and Christopher D. Manning. Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1556–1566. Association for Computational Linguistics, Beijing, China, **2015**.

[171] Inigo Lopez-Gazpio, Montse Maritxalar, Mirella Lapata, and Eneko Agirre. Word n-gram attention models for sentence similarity and inference. *Expert Systems with Applications*, 132:1–11, **2019**.

[172] Samuel Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642. **2015**.

[173] Chen Qu, Liu Yang, Minghui Qiu, W Bruce Croft, Yongfeng Zhang, and Mohit Iyyer. BERT with history answer embedding for conversational question answering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1133–1136. **2019**.

[174] Sofian Chaybouti, Achraf Saghe, and Aymen Shabou. EfficientQA: a RoBERTa Based Phrase-Indexed Question-Answering System. *CoRR*, abs/2101.02157, **2021**.

[175] Da Yin, Tao Meng, and Kai-Wei Chang. SentiBERT: A Transferable Transformer-Based Architecture for Compositional Sentiment Semantics. In

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 3695–3706. **2020**.

[176] Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. On the Sentence Embeddings from BERT for Semantic Textual S0imilarity. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9119–9130. **2020**.

[177] Xingyi Cheng. Dual-View Distilled BERT for Sentence Embedding. *CoRR*, abs/2104.08675, **2021**.

[178] Tingyu Xia, Yue Wang, Yuan Tian, and Yi Chang. Using Prior Knowledge to Guide BERT's Attention in Semantic Textual Matching Tasks. In *Proceedings of the Web Conference 2021*, pages 2466–2475. **2021**.

[179] Yuanmeng Yan, Rumei Li, Sirui Wang, Fuzheng Zhang, Wei Wu, and Weiran Xu. ConSERT: A Contrastive Framework for Self-Supervised Sentence Representation Transfer. *CoRR*, abs/2105.11741, **2021**.

[180] Konstantinos Bougiatiotis and Theodoros Giannakopoulos. Enhanced movie content similarity based on textual, auditory and visual information. *Expert Systems with Applications*, 96:86–102, **2018**.

[181] Abdullah M Sheneamer. Multiple Similarity-based Features Blending for Detecting Code Clones using Consensus-Driven Classification. *Expert Systems with Applications*, 183:115364, **2021**.

[182] Chensi Cao, Feng Liu, Hai Tan, Deshou Song, Wenjie Shu, Weizhong Li, Yiming Zhou, Xiaochen Bo, and Zhi Xie. Deep learning and its applications in biomedicine. *Genomics, proteomics & bioinformatics*, 16(1):17–32, **2018**.

[183] Ermal Elbasani and Jeong-Dong Kim. AMR-CNN: Abstract Meaning Representation with Convolution Neural Network for Toxic Content Detection. *Journal of Web Engineering*, pages 677–692, **2022**.

[184] Chuhan Wu, Fangzhao Wu, Sixing Wu, Junxin Liu, Zhigang Yuan, and Yongfeng Huang. Thu_NGN at SemEval-2018 Task 3: Tweet Irony Detection with Densely Connected LSTM and Multi-task Learning. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 51–56. **2018**.

[185] Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. How to fine-tune BERT for text classification? In *China national conference on Chinese computational linguistics*, pages 194–206. Springer, **2019**.

[186] Shaomin Zheng and Meng Yang. A new method of improving BERT for text classification. In *International Conference on Intelligent Science and Big Data Engineering*, pages 442–452. Springer, **2019**.

[187] Cynthia Van Hee, Els Lefever, and Véronique Hoste. SemEval-2018 Task 3: Irony Detection in English Tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 39–50. **2018**.

[188] Alessandra Teresa Cignarella, Simona Frenda, Valerio Basile, Cristina Bosco, Viviana Patti, and Paolo Rosso. Overview of the EVALITA 2018 Task on Irony Detection in Italian Tweets (IronITA). In *Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2018)*, volume 2263, pages 1–6. CEUR-WS, **2018**.

[189] José-Ángel González, Lluís-Felip Hurtado, and Ferran Pla. ELiRF-UPV at IroSvA: Transformer Encoders for Spanish Irony Detection. In *IberLEF@ SEPLN*, pages 278–284. **2019**.

[190] Farah Benamara, Cyril Grouin, Jihen Karoui, Véronique Moriceau, and Isabelle Robba. Analyse d'opinion et langage figuratif dans des tweets: présentation et résultats du Défi Fouille de Textes DEFT2017. **2017**.

[191] Andrea Cimino, Lorenzo De Mattei, and Felice Dell'Orletta. Multi-task Learning in Deep Neural Networks at Evalita 2018. *Proceedings of the 6th*

evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA'18), pages 86–95, **2018**.

[192] Felipe Bravo-Marquez, Marcelo Mendoza, and Barbara Poblete. Meta-level sentiment models for big social data analysis. *Knowledge-based systems*, 69:86–99, **2014**.

[193] Mike Thelwall, Kevan Buckley, and Georgios Paltoglou. Sentiment strength detection for the social web. *Journal of the American Society for Information Science and Technology*, 63(1):163–173, **2012**.

[194] José Raniery Ferreira, Diego Armando Cardona Cardenas, Ramon Alfredo Moreno, Marina de Fátima de Sá Rebelo, José Eduardo Krieger, and Marco Antonio Gutierrez. Multi-view Ensemble Convolutional Neural Network to Improve Classification of Pneumonia in Low Contrast Chest X-Ray Images. In *2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, pages 1238–1241. IEEE, **2020**.

[195] Hande Taslioglu and Pinar Karagoz. Irony Detection on Microposts with Limited Set of Features. In *Proceedings of the Symposium on Applied Computing*, pages 1076–1081. **2017**.

[196] Oğuzhan Dülger. Türkçe Metinlerde ironi Tespiti. *Gazi Üniversitesi, Erişim Tarihi*, 3:2020, **2018**.

[197] Han Ren, Yafeng Ren, Xia Li, Wenhe Feng, and Maofu Liu. Natural logic inference for emotion detection. In *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data*, pages 424–436. Springer, **2017**.

[198] Donglei Tang, Zhikai Zhang, Yulan He, Chao Lin, and Deyu Zhou. Hidden topic–emotion transition model for multi-level social emotion detection. *Knowledge-Based Systems*, 164:426–435, **2019**.

[199] Dong Zhang, Xincheng Ju, Junhui Li, Shoushan Li, Qiaoming Zhu, and Guodong Zhou. Multi-modal Multi-label Emotion Detection with Modality and Label Dependence. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3584–3593. **2020**.

[200] Rui Wang, Andrew Finch, Masao Utiyama, and Eiichiro Sumita. Sentence embedding for neural machine translation domain adaptation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 560–566. **2017**.

[201] Christos Baziotis, Nikos Athanasiou, Alexandra Chronopoulou, Athanasia Kolovou, Georgios Paraskevopoulos, Nikolaos Ellinas, Shrikanth Narayanan, and Alexandros Potamianos. NTUA-SLP at SemEval-2018 Task 1: Predicting Affective Content in Tweets with Deep Attentive RNNs and Transfer Learning, **2018**.

[202] Grace Gee and Eugene Wang. psyML at SemEval-2018 Task 1: Transfer learning for sentiment and emotion analysis. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 369–376. **2018**.

[203] Yanghoon Kim, Hwanhee Lee, and Kyomin Jung. AttnConvnet at SemEval-2018 Task 1: Attention-based Convolutional Neural Networks for Multi-label Emotion Classification, **2018**.

[204] Ben Eisner, Tim Rocktäschel, Isabelle Augenstein, Matko Bošnjak, and Sebastian Riedel. emoji2vec: Learning Emoji Representations from their Description, **2016**.

[205] Nourah Alswaidan and Mohamed El Bachir Menai. A survey of state-of-the-art approaches for emotion recognition in text. *Knowledge and Information Systems*, pages 1–51, **2020**.

[206] Iqra Ameer, Noman Ashraf, Grigori Sidorov, and Helena Gómez Adorno. Multi-label emotion classification using content-based features in Twitter. *Computación y Sistemas*, 24(3):1159–1164, **2020**.

[207] Fei Ding, Xin Kang, Shun Nishide, Zhijin Guan, and Fuji Ren. A fusion model for multi-label emotion classification based on BERT and topic clustering. In *International Symposium on Artificial Intelligence and Robotics 2020*, volume 11574, pages 98–111. SPIE, **2020**.

[208] Iqra Ameer, Grigori Sidorov, Helena Gomez-Adorno, and Rao Muhammad Adeel Nawab. Multi-Label Emotion Classification on Code-Mixed Text: Data and Methods. *IEEE Access*, 10:8779–8789, **2022**.

[209] Saif Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. SemEval-2018 Task 1: Affect in tweets. In *Proceedings of the 12th international workshop on semantic evaluation*, pages 1–17. **2018**.

[210] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, **2013**.

[211] Hardik Meisheri and Lipika Dey. TCS research at SemEval-2018 Task 1: Learning robust representations using multi-attention architecture. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 291–299. **2018**.

[212] Alec Radford, Rafal Józefowicz, and Ilya Sutskever. Learning to Generate Reviews and Discovering Sentiment. *CoRR*, abs/1704.01444, **2017**.

[213] S Rajalakshmi, S Milton Rajendram, TT Mirnalinee, et al. SSN MLRG1 at SemEval-2018 Task 1: Emotion and sentiment intensity detection using rule based feature selection. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 324–328. **2018**.

[214] Prabod Rathnayaka, Supun Abeysinghe, Chamod Samarajeewa, Isura Manchanayake, Malaka J. Walpola, Rashmika Nawaratne, Tharindu Bandaragoda, and Damminda Alahakoon. Gated Recurrent Neural Network Approach for Multilabel Emotion Detection in Microblogs, **2019**.

[215] Sungjoon Park, Jiseon Kim, Seonghyeon Ye, Jaeyeol Jeon, Hee Young Park, and Alice Oh. Dimensional emotion detection from categorical emotion. *arXiv preprint arXiv:1911.02499*, **2019**.

[216] Lajanugen Logeswaran and Honglak Lee. An efficient framework for learning sentence representations. *CoRR*, abs/1803.02893, **2018**.

[217] Alexis Conneau and Douwe Kiela. SentEval: An Evaluation Toolkit for Universal Sentence Representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. **2018**.

[218] Hariom A. Pandya and Brijesh S. Bhatt. Question Answering Survey: Directions, Challenges, Datasets, Evaluation Matrices. *CoRR*, abs/2112.03572, **2021**.

[219] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ Questions for Machine Comprehension of Text. *CoRR*, abs/1606.05250, **2016**.

[220] Rudolf Kadlec, Martin Schmid, Ondřej Bajgar, and Jan Kleindienst. Text Understanding with the Attention Sum Reader Network. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 908–918. **2016**.

[221] Yiming Cui, Zhipeng Chen, Si Wei, Shijin Wang, Ting Liu, and Guoping Hu. Attention-over-Attention Neural Networks for Reading Comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 593–602. **2017**.

[222] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A lite BERT for Self-supervised Learning of Language Representations. *CoRR*, abs/1909.11942, **2019**.

[223] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. ELECTRA: PRE-TRAINING TEXT ENCODERS AS DISCRIMINATORS RATHER THAN GENERATORS. *ELECTRA*, 85:90, **2016**.

[224] Lianzhe Huang, Xin Sun, Sujian Li, Linhao Zhang, and Houfeng Wang. Syntax-aware graph attention network for aspect-level sentiment classification. In *Proceedings of the 28th international conference on computational linguistics*, pages 799–810. **2020**.

[225] Shuo Liang, Wei Wei, Xian-Ling Mao, Fei Wang, and Zhiyong He. BiSyn-GAT: Bi-Syntax Aware Graph Attention Network for Aspect-based Sentiment Analysis. In *Findings of the Association for Computational Linguistics: ACL 2022*. Association for Computational Linguistics, **2022**.

[226] Syntax-guided text generation via graph neural network, author=Guo, Qipeng and Qiu, Xipeng and Xue, Xiangyang and Zhang, Zheng, journal=Science China Information Sciences, volume=64, number=5, pages=1–10, year=2021, publisher=Springer.

[227] Michael Sejr Schlichtkrull, Nicola De Cao, and Ivan Titov. Interpreting Graph Neural Networks for NLP With Differentiable Edge Masking. *CoRR*, abs/2010.00577, **2020**.

[228] Alireza Mohammadshahi and James Henderson. Syntax-Aware Graph-to-Graph Transformer for Semantic Role Labelling. *CoRR*, abs/2104.07704, **2021**.

[229] Jasmijn Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Sima'an. Graph Convolutional Encoders for Syntax-aware Neural Machine Translation. *CoRR*, abs/1704.04675, **2017**.

[230] Attila Nagy, Patrick Nanys, Balázs Frey Konrád, Bence Bial, and Judit Ács. Syntax-based data augmentation for Hungarian-English machine translation. *CoRR*, abs/2201.06876, **2022**.

[231] Jiangang Bai, Yujing Wang, Yiren Chen, Yaming Yang, Jing Bai, Jing Yu, and Yunhai Tong. Syntax-BERT: Improving Pre-trained Transformers with Syntax Trees. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3011–3020. **2021**.

[232] Diego Marcheggiani and Ivan Titov. Graph Convolutions over Constituent Trees for Syntax-Aware Semantic Role Labeling. *CoRR*, abs/1909.09814, **2019**.

[233] Xuan-Phi Nguyen, Shafiq R. Joty, Steven C. H. Hoi, and Richard Socher. Tree-structured Attention with Hierarchical Accumulation. *CoRR*, abs/2002.08046, **2020**.

[234] Sufeng Duan, Hai Zhao, Dongdong Zhang, and Rui Wang. Syntax-aware data augmentation for neural machine translation. *arXiv preprint arXiv:2004.14200*, **2020**.

[235] Defeng Xie, Jianmin Ji, Jiafei Xu, and Ran Ji. Combining Improvements for Exploiting Dependency Trees in Neural Semantic Parsing. In *Pacific Rim International Conference on Artificial Intelligence*, pages 58–72. Springer, **2021**.

[236] Longchao Gong, Yan Li, Junjun Guo, Zhengtao Yu, and Shengxiang Gao. Enhancing low-resource neural machine translation with syntax-graph guided self-attention. *Knowledge-Based Systems*, 246:108615, **2022**.

[237] Emanuele Bugliarello and Naoaki Okazaki. Improving Neural Machine Translation with Parent-Scaled Self-Attention. *CoRR*, abs/1909.03149, **2019**.

[238] Devendra Singh Sachan, Yuhao Zhang, Peng Qi, and William L. Hamilton. Do Syntax Trees Help Pre-trained Transformers Extract Information? *CoRR*, abs/2008.09084, **2020**.

[239] Usman Ahmed, Lubna Zafar, Faiza Qayyum, and Muhammad Arshad Islam. Irony Detector at SemEval-2018 Task 3: Irony Detection in English Tweets using Word Graph. In *Proceedings of the 12th international workshop on semantic evaluation*, pages 581–586. **2018**.

[240] Mrinmaya Sachan and Eric Xing. Machine comprehension using rich semantic representations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 486–492. **2016**.

[241] Boris Galitsky. Employing Abstract Meaning Representation to Lay the Last-Mile Toward Reading Comprehension. In *Artificial Intelligence for Customer Relationship Management*, pages 57–86. Springer, **2020**.

[242] Sho Takase, Jun Suzuki, Naoaki Okazaki, Tsutomu Hirao, and Masaaki Nagata. Neural headline generation on abstract meaning representation. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 1054–1059. **2016**.

[243] Zhuosheng Zhang, Yuwei Wu, Hai Zhao, Zuchao Li, Shuailiang Zhang, Xi Zhou, and Xiang Zhou. Semantics-aware BERT for language understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9628–9635. **2020**.

[244] Ying Qin and Ye Liang. Semantic Analysis and Evaluation of Translation Based on Abstract Meaning Representation. In *International Conference on Web Information Systems and Applications*, pages 268–275. Springer, **2020**.

[245] Aviv Slobodkin, Leshem Choshen, and Omri Abend. Semantics-aware Attention Improves Neural Machine Translation. *CoRR*, abs/2110.06920, **2021**.

[246] Changmao Li and Jeffrey Flanigan. Improving Neural Machine Translation with the Abstract Meaning Representation by Combining Graph and Sequence Transformers. In *Proceedings of the 2nd Workshop on Deep Learning on Graphs for Natural Language Processing (DLG4NLP 2022)*, pages 12–21. **2022**.

[247] Hardy and Andreas Vlachos. Guided Neural Language Generation for Abstractive Summarization using Abstract Meaning Representation. *CoRR*, abs/1808.09160, **2018**.

[248] Panagiotis Kouris, Georgios Alexandridis, and Andreas Stafylopatis. Text summarization based on semantic graphs: An abstract meaning representation graph-to-text deep learning approach. **2022**.

[249] Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. Effective Approaches to Attention-based Neural Machine Translation. *CoRR*, abs/1508.04025, **2015**.

[250] Hanqi Jin, Tianming Wang, and Xiaojun Wan. Semsum: Semantic dependency guided neural abstractive summarization. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8026–8033. **2020**.

[251] Leonardo F. R. Ribeiro, Mengwen Liu, Iryna Gurevych, Markus Dreyer, and Mohit Bansal. FactGraph: Evaluating Factuality in Summarization with Semantic Graph Representations, **2022**.

[252] Shaowei Yao, Tianming Wang, and Xiaojun Wan. Heterogeneous graph transformer for graph-to-sequence learning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7145–7154. **2020**.

[253] Jungwoo Lim, Dongsuk Oh, Yoonna Jang, Kisu Yang, and Heuiseok Lim. I Know What You Asked: Graph Path Learning using AMR for Commonsense Reasoning. *CoRR*, abs/2011.00766, **2020**.

[254] Tao Liu, Xin Wang, Chengguo Lv, Ranran Zhen, and Guohong Fu. Sentence matching with syntax-and semantics-aware BERT. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3302–3312. **2020**.

[255] Ziyi Shou, Yuxin Jiang, and Fangzhen Lin. AMR-DA: Data Augmentation by Abstract Meaning Representation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3082–3098. **2022**.

[256] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546 vol. 1. **2005**.

[257] Richard Socher, Christopher D Manning, and Andrew Y Ng. Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *Proceedings of the NIPS-2010 deep learning and unsupervised feature learning workshop*, volume 2010, pages 1–9. **2010**.

[258] Susan Craw. *Manhattan Distance*, pages 790–791. Springer US, Boston, MA, **2017**. ISBN 978-1-4899-7687-1.

[259] Wen-tau Yih, Kristina Toutanova, John C Platt, and Christopher Meek. Learning Discriminative Projections for Text Similarity Measures. In *Proceedings of the fifteenth conference on computational natural language learning*, pages 247–256. **2011**.

[260] Jonas Mueller and Aditya Thyagarajan. Siamese Recurrent Architectures for Learning Sentence Similarity. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30. **2016**.

[261] Elvys Linhares Pontes, Stéphane Huet, Andréa Carneiro Linhares, and Juan-Manuel Torres-Moreno. Predicting the Semantic Textual Similarity with Siamese CNN and LSTM. *CoRR*, abs/1810.10641, **2018**.

[262] Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*. European Language Resources Association (ELRA), Reykjavik, Iceland, **2014**.

[263] Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. SemEval-2017 Task 1: Semantic Textual Similarity Multilingual and Crosslingual Focused Evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14. **2017**.

[264] Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177. **2004**.

[265] Bo Pang and Lillian Lee. A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 271–278. **2004**.

[266] Janyce Wiebe, Theresa Wilson, and Claire Cardie. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39(2):165–210, **2005**.

[267] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642. **2013**.

[268] William B Dolan, Chris Quirk, and Chris Brockett. Unsupervised Construction of Large Paraphrase Corpora: Exploiting Massively Parallel News Sources.

In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 350–356. **2004**.

[269] Nils Reimers, Philip Beyer, and Iryna Gurevych. Task-oriented intrinsic evaluation of semantic textual similarity. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 87–96. **2016**.

[270] Adina Williams, Nikita Nangia, and Samuel Bowman. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. **2018**.

[271] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The Graph Neural Network Model. *IEEE transactions on neural networks*, 20(1):61–80, **2008**.

[272] Weijing Shi and Raj Rajkumar. Point-GNN: Graph Neural Network for 3D Object Detection in a Point Cloud. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1711–1719. **2020**.

[273] Difei Gao, Ke Li, Ruiping Wang, Shiguang Shan, and Xilin Chen. Multi-Modal Graph Neural Network for Joint Reasoning on Vision and Scene Text. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12746–12756. **2020**.

[274] Abdullah Hamid, Nasrullah Sheikh, Naina Said, Kashif Ahmad, Asma Gul, Laiq Hassan, and Ala I. Al-Fuqaha. Fake News Detection in Social Media using Graph Neural Networks and NLP Techniques: A COVID-19 Use-case. *CoRR*, abs/2012.07517, **2020**.

[275] Wenxiong Liao, Bi Zeng, Jianqi Liu, Pengfei Wei, Xiaochun Cheng, and Weiwen Zhang. Multi-level Graph Neural Network for Text Sentiment Analysis. *Computers & Electrical Engineering*, 92:107096, **2021**.

[276] Liang Yao, Chengsheng Mao, and Yuan Luo. Graph Convolutional Networks for Text Classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 7370–7377. **2019**.

[277] Yuhao Zhang, Peng Qi, and Christopher D. Manning. Graph Convolution over Pruned Dependency Trees Improves Relation Extraction. *CoRR*, abs/1809.10185, **2018**.

[278] Adam Roberts, Colin Raffel, and Noam Shazeer. How Much Knowledge Can You Pack Into the Parameters of a Language Model? *CoRR*, abs/2002.08910, **2020**.

[279] Matthew Honnibal and Ines Montani. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. *To appear*, 7(1):411–420, **2017**.

[280] Timothy Dozat and Christopher D. Manning. Deep Biaffine Attention for Neural Dependency Parsing. *CoRR*, abs/1611.01734, **2016**.

[281] Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. Stanza: A Python Natural Language Processing Toolkit for Many Human Languages. *CoRR*, abs/2003.07082, **2020**.

[282] Shiyao Wang, Minlie Huang, Zhidong Deng, et al. Densely Connected CNN with Multi-scale Feature Attention for Text Classification. In *IJCAI*, pages 4468–4474. **2018**.

[283] Hao Peng, Jianxin Li, Yu He, Yaopeng Liu, Mengjiao Bao, Lihong Wang, Yangqiu Song, and Qiang Yang. Large-Scale Hierarchical Text Classification with Recursively Regularized Deep Graph-CNN. In *Proceedings of the 2018 world wide web conference*, pages 1063–1072. **2018**.

[284] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks, **2017**.

[285] Li Yuan, Jin Wang, Liang-Chih Yu, and Xuejie Zhang. Graph Attention Network with Memory Fusion for Aspect-level Sentiment Analysis. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 27–36. **2020**.

[286] Kai Wang, Weizhou Shen, Yunyi Yang, Xiaojun Quan, and Rui Wang. Relational Graph Attention Network for Aspect-based Sentiment Analysis. *CoRR*, abs/2004.12362, **2020**.

[287] Valerio Basile, Andrea Bolioli, Viviana Patti, Paolo Rosso, and Malvina Nissim. Overview of the Evalita 2014 SENTIment POlarity Classification Task. *Overview of the Evalita 2014 SENTIment POLarity Classification Task*, pages 50–57, **2014**.

[288] Ji Ho Park, Peng Xu, and Pascale Fung. PlusEmo2Vec at SemEval-2018 Task 1: Exploiting emotion knowledge from emoji and #hashtags, **2018**.

[289] Li Zhang, Tao Xiang, and Shaogang Gong. Learning a deep embedding model for zero-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2021–2030. **2017**.

[290] Baosong Yang, Zhaopeng Tu, Derek F Wong, Fandong Meng, Lidia S Chao, and Tong Zhang. Modeling Localness for Self-Attention Networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4449–4458. **2018**.

[291] Thanh-Tung Nguyen, Xuan-Phi Nguyen, Shafiq Joty, and Xiaoli Li. Differentiable Window for Dynamic Local Attention. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6589–6599. **2020**.

[292] Alexis Conneau, Germán Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. What you can cram into a single $ &!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136. **2018**.

[293] Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. What does BERT learn about the structure of language? In *ACL 2019-57th Annual Meeting of the Association for Computational Linguistics*. **2019**.

[294] Wanxiang Che, Yijia Liu, Yuxuan Wang, Bo Zheng, and Ting Liu. Towards Better UD Parsing: Deep Contextualized Word Embeddings, Ensemble, and Treebank Concatenation. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 55–64. **2018**.

[295] Daniel Kondratyuk. 75 Languages, 1 Model: Parsing Universal Dependencies Universally. *CoRR*, abs/1904.02099, **2019**.

[296] Milan Straka, Jana Straková, and Jan Hajič. Evaluating Contextualized Embeddings on 54 Languages in POS Tagging, Lemmatization and Dependency Parsing, **2019**.

8. APPENDIX

8.1. UCCA guideline with Turkish examples

A text in the foundational layer of UCCA representation consists of `Scenes`. It may consist of one or more scenes as shown in the following examples.

- Ahmet okula gitti (in English, *"Ahmet went to the school"*) (1 Scene)
- Ahmet eve döndü ve duş aldı (in English, *"Ahmet went back home and took a shower"*) (2 Scene)

8.1.1. Categories

8.1.1.1. Scene Elements: Each `Scene` should have a main relation describing a movement or action called `Process` (P) or a temporally persistent state called `State` (S). The other components of a `Scene` are `Participant` (A), which can be one or more, and `Adverbial` (D), which describes the relation in time, location or ground. Below are examples from Turkish of Scene-elements.

- Ben $\langle$bir tutsağım$\rangle_S$ (in English, *"I am a prisoner"*)
- Ahmet okula yürüyerek $\langle$gitti$\rangle_P$ (in English, *"Ahmet went to school on foot"*)
- $\langle$Elmayı$\rangle_A$ alabilir (in English, *"He may take the apple"*)
- $\langle$Ayşe$\rangle_A$ $\langle$okulda$\rangle_A$ kaldı (in English, *"Ayşe stayed at school"*)
- Ahmet yüzmeye $\langle$başladı$\rangle_D$ (in English, *"He started swimming"*)
- Soruyu $\langle$hızlıca$\rangle_D$ cevapladı (in English, *"She answered the question quickly"*)
- Ayşe $\langle$sık sık$\rangle_D$ spora gider (in English, *"Ayşe often goes to the gym"*)

8.1.1.2. Non-Scene Unit Elements: Non-Scene relations do not evoke a `Scene`, which is the main difference with the Scene elements. The main concept in non-Scene units is `Center` (C), and other relations that detail the `Centers`. While `Elaborator` (E) determines the semantic type or quantification of the magnitude of the parent entity that is a `Center`, `Connector` (N) connects entities that have similar features or types. Finally, `Relator` (R) relates an entity to other relations or units that are attached with different aspects.

- $\langle$1996$\rangle_C$ $\langle$yılı$\rangle_E$ (in English, *"the year 1966"*)
- $\langle$onun$\rangle_E$ $\langle$eli$\rangle_C$ (in English, *"his hand"*)
- $\langle$biraz$\rangle_E$ $\langle$şeker$\rangle_C$ (in English, *"some sugar"*)
- $\langle\langle$Ben$\rangle_C$ $\langle$ve$\rangle_N$ $\langle\langle$(benim)$\rangle_{E-IMPLICIT}$ $\langle$arkadaşım$\rangle_C\rangle_C$ okula beraber gittik (in English, *"I and my friend went to school together"*)
- Ali [$\langle$fırının$\rangle_C$ $\langle$içindeki$\rangle_R$] kurabiyeleri] aldı (in English, *"Ali took the cookies from the oven"*)

8.1.1.3. Inter-Scene Relations: Inter-Scene relations category is composed of `Parallel Scene` (H), `Linker` (L) and `Ground` (G). `Parallel Scene` is a `Scene` that does not take place in the main Scene as a `Participant`, a `Center`, or an `Elaborator`. The `Parallel Scenes` can be linked to other Scenes with `Linker`, which is a relational word between `Parallel Scenes`. `Ground` is a unit that relates units to their speech event, which can be either a speaker, or a hearer. The main difference with `Linker` is that it does not relate Scenes. Linkage is a term used in Inter-Scene relations in which a `Scene` is a unit in one of `Participant`, `Center`, `Elaborator`, `Adverbial` (described in Section **??**) or `Parallel Scene`.

- $\langle$Eğer$\rangle_L$ $\langle$okula gidersen$\rangle_H$ $\langle$Ahmet ile karşılaşırsın$\rangle_H$ (in English, *"If you go to school, you will meet Ahmet"*)

- $\langle$Arkadaşını beklerken$\rangle_H$ $\langle$ayakkabısını boyadı$\rangle_H$ (in English, *"While waiting for her friend, she polished her shoes"*)
- $\langle$Sadece kendi istediklerini söyledin$\rangle_H$ $\langle$çünkü$\rangle_L$ $\langle$sen de suçlusun$\rangle_H$ (in English, *"You just said what you wanted because you're guilty too"*)
- $\langle$İlginçtir$\rangle_G$ okumakta zorlanmadı (in English, *"Interestingly, it wasn't hard to read"*)
- $\langle$Gördüğünüz gibi$\rangle_H$ $\langle$gelmediler$\rangle_H$ (in English, *"They didn't come as you see"*)
- $\langle$Eski kocası$\rangle_A$ her zaman oradadır (in English, *"Her/his ex husband is always there"*)
- $\langle$Seni üzmekten$\rangle_A$ korkuyorum (in English, *"I'm afraid to upset you"*)
- $\langle\langle$Her$\rangle_E$ $\langle$istediğini$\rangle_P\rangle_C$ yerine getiriyordum. (in English, *"I was doing whatever s/he want"*)
- Ürkütücü şeyler $\langle\langle$bu$_E$ $\langle$anlattıklarınız$\rangle_P\rangle_C$ (in English, *"These are the scary things you're talking about"*)
- $\langle$Dar yollarda koşarak giden$\rangle_D$ Kerem'i yakaladım (in English, *"I caught Kerem running on narrow roads"*)
- $\langle$Bahçeye giren (köpek)$\rangle_E$ köpek kahverengidir (in English, *"The dog entering the garden is brown"*)
- $\langle$[Yan daireye] taşınan (Ahmet)$\rangle_E$ Ahmet evime geldi (in English, *"Ahmet who moved to the next flat, came to my house"*)

8.1.1.4. Other: The final category is Other in which the `Function` (F) unit is only a part of the construction.

- $\langle$Ayy$\rangle_F$ sandalyeden düştü (in English, *"Ouch he fell from the chair"*)
- İstanbul $\langle$'a$\rangle_F$ mı gidiyorsun (in English, *"Your are going to the Erkekler Park"*)
- Kerem $\langle$bir$\rangle_F$ an durdu (in English, *"Kerem stopped for a moment"*)

8.1.1.5. Remote and Implicit Units:

If an entity is missing and referred from another position in the text, an edge is added for the entity as `IMPLICIT`. If it is not referred in the text, a new token is created as a `REMOTE` unit.

- $\langle$(O)$\rangle_{A-IMPLICIT}$ okula gelmedi (in English, *"He didn't come to school"*)
- [$\langle$(Benim)$\rangle_{E-IMPLICIT}$ Çocukluğum] aklıma geldi (in English, *"I remembered my childhood"*)
- [Ali okuldan geldi]$_H$ ve [televizyon izledi $\langle$(Ali)$\rangle_{A-REMOTE}$] (in English, *"Ali came from school and watched television"*)
- [Okula yeni kayıt olan $\langle$(çocuk)$\rangle_{A-REMOTE}$] çocuk] bugün gelmedi (in English, *"The newly enrolled child did not come today"*)
- Ne [ondan bahsedebildim] ne [yaşadıklarımdan $\langle$(bahsedebildim)$\rangle_{P-REMOTE}$] (in English, *"I could neither talk about her/his nor talk about my experiences"*)

8.2. Hyperparameters of Text Classification

8.2.1. Irony Detection

Table 8.1 lists the hyperparameter values used in the models.

Parameters	**Turkish Irony Dataset [8]**			**IronyTR Dataset [9]**		
	UCCA-CNN	UCCA-GCN	UCCA GAT	UCCA-CNN	UCCA-GCN	UCCA GAT
weight decay	0.1	0.2	0.2	0.2	0.2	0.2
batch size	64	2	1	64	2	1
learning rate	0.0001	0.005	0.0005	0.001	0.0001	0.0005
filter sizes	[3,4,5]	-	-	[3,4,5,6]	-	-
# of filters	50	-	-	50	-	-
dropout rate	0.2	0.1	0.1	0.2	0.2	0.2
# of MLP	2	-	-	2	-	-
#of neurons ins in FC	200	-	-	200	-	-
# of hidden		200	800		200	800
# of head	-	-	1	-	-	1

Table 8.1 Hyperparameters used for the different models in experiments

8.2.2. Multi-label Emotion Classification

Table 8.2 lists the hyperparameter values used in the model.

Parameters	**SemEval-2018 Dataset [10]**		**GoEmotions Dataset [6]**	
	UCCA-GAT	Dep-GAT	UCCA-GAT	Dep-GAT
weight decay	0.1	0.2	0.2	0.2
batch size	1	1	1	1
learning rate	0.001	0.005	0.001	0.005
dropout rate	0.2	0.1	0.1	0.2
# of hidden	400	400	400	400
# of head	2	4	2	4

Table 8.2 Hyperparameters used for the different models in monolingual experiments

8.2.3. Transfer Downstream Tasks

Table 8.3, 8.4, and 8.5 list the hyperparameter values used in the UCCA-GAT, AMR-GAT and Dep-GAT models, respectively, for transfer downstream tasks.

Parameters	MR	CR	SUBJ	MPQA	SST-2	TREC	MRPC
weight decay	0.2	0.1	0.2	0.2	0.2	0.1	0.1
batch size	1	1	1	1	1	1	1
learning rate	1e-4	1e-4	1e-4	1e-4	1e-4	1e-4	1e-4
dropout rate	0.1	0.1	0.1	0.2	0.1	0.1	0.1
# of hidden	800	800	800	800	400	800	800
# of head	2	1	2	2	4	1	1

Table 8.3 Hyperparameters used for the UCCA-GAT for transfer tasks in experiments

Parameters	MR	CR	SUBJ	MPQA	SST-2	TREC	MRPC
weight decay	0.1	0.1	0.2	0.1	0.2	0.1	0.1
batch size	1	1	1	1	1	1	1
learning rate	1e-4	1e-4	1e-4	1e-4	1e-4	1e-4	1e-4
dropout rate	0.2	0.1	0.2	0.1	0.2	0.1	0.1
# of hidden	800	400	800	800	800	400	800
# of head	2	1	2	2	4	1	1

Table 8.4 Hyperparameters used for the AMR-GAT for transfer tasks in experiments

Parameters	MR	CR	SUBJ	MPQA	SST-2	TREC	MRPC
weight decay	0.1	0.1	0.2	0.1	0.2	0.1	0.1
batch size	1	1	1	1	1	1	1
learning rate	2e-5	2e-5	2e-5	2e-5	2e-5	2e-5	2e-5
dropout rate	0.1	0.1	0.2	0.1	0.2	0.1	0.1
# of hidden	800	400	800	800	800	400	800
# of head	2	1	2	2	4	1	1

Table 8.5 Hyperparameters used for the Dep-GAT for transfer tasks in experiments