ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

ABSTRACT MEANING REPRESENTATION OF TURKISH

M.Sc. THESIS

Kadriye Elif ORAL

Department of Computer Engineering

Computer Engineering Programme

NOVEMBER 2022

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

ABSTRACT MEANING REPRESENTATION OF TURKISH

M.Sc. THESIS

Kadriye Elif ORAL
(5041915266)

Department of Computer Engineering

Computer Engineering Programme

Thesis Advisor: Assoc. Prof. Dr. Gulsen Eryigit

NOVEMBER 2022

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

TÜRKÇENİN SOYUT ANLAM TEMSİLLERİ

YÜKSEK LİSANS TEZİ

Kadriye Elif ORAL
(5041915266)

Bilgisayar Mühendisliği Anabilim Dalı

Bilgisayar Mühendisliği Programı

Tez Danışmanı: Doç. Dr. Gülşen Eryiğit

KASIM 2022

Kadriye Elif ORAL, a M.Sc. student of ITU Graduate School student ID 5041915266 successfully defended the thesis entitled "ABSTRACT MEANING REPRESENTATION OF TURKISH", which he/she prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Assoc. Prof. Dr. Gulsen Eryigit** Istanbul Technical University	.....
Jury Members :	**Assoc. Prof Dr. Cuneyd Tantug** Istanbul Technical University	.....
	Asst. Prof. Dr. Gozde Gul Sahin Koc University	.....

Date of Submission : **22 October 2022**
Date of Defense : **12 November 2022**

To Lokum, Venüs and Limon

FOREWORD

I would like to express my deepest gratitude to my advisor, Assoc. Dr. Gülşen Eryiğit, for her sincerity, guidance and encouragement. She has been a teacher, a leader, and a role model to me and has always inspired me on my journey through this master's degree. I always consider myself very fortunate that our paths crossed. This thesis would not have been possible without her, as she showed me the right direction from the first step to the end of this journey.

Also, I am deeply grateful to Ali Acar. He is a great research colleague and friend to whom I am very grateful for his great help and kind support. I am also very grateful to Zahra Azin for introducing me to AMR. She is the true definition of diligence and determination. My special thanks go to Tolga Kalaycı. He is not only a manager, but also a great mentor and a friend. I am grateful to him for giving me the opportunity to complete this thesis and for his constructive feedback, kind support, and advice during my master's studies that helped me grow professionally. I would also like to thank my teammates, Semih, Furkan, Fırat, Ahmet and Yusuf . They have shown me great understanding, support and friendship. They turned my job into fun so that I was able to balance my professional and academic life.

Finally, I am extremely grateful to my family, whose constant love and support has motivated me and always stood by my side. It has been a long journey with many ups and downs. Whenever I felt tired, they reminded me what was important in life. They always believed in me. I owe my deepest gratitude to Bilgin, my great love. I am forever grateful for his unconditional love and patience throughout the thesis process and every day.

November 2022

Kadriye Elif ORAL
(Engineer)

TABLE OF CONTENTS

ABBREVIATIONS

NLP : Natural Language Processing
SAT : Soyut Anlam Temsili
MR : Meaning Representation
ML : Machine Learning
AMR : Abstract Meaning Representation
DRS : Discourse Representation Structures
GMB : Groningen Meaning Bank
UCCA : Universal Conceptual Cognitive Annotation
UMR : Uniform Meaning Representation
DAG : Directed Acyclic Graph
DS : Derivational Suffix
IS : Inflectional Suffix
HPS : Highly Productive Suffix
HPV : Verb derived Highly Productive suffixes
IMST : Itu-Metu-Sabanci-Treebank
tLP : The Little Prince
IAA : Inter-annotated Agreement
RBP : Rule-based Parser
POS : Part-of-Speech
DDP : Data-driven Parser
RBP : Rule-based Parser
MRL : Morphologically Rich Languages
TC : Type Classifier
PC : Person Classifier
PTC : Person/Thing Classifier
AbsC : Abstract Classifier
IC : Invoke Classifier
ConjC : Conjunction Classifier
CauC : Cause Classifier
QC : Question Classifier
NeC : Named Entity Classifier
ArgC : Argument Classifier
nonCC : Non-Core Classifier
constC : Constant Classifier
SMOTE : Synthetic Minority Oversampling Technique
NER : Named Entity Recognizer
SRL : Semantic Role Labeling
BERT : Bidirectional Encoder Representations from Transformers
ELECTRA : Efficiently Learning an Encoder that Classifies Token Replacements Accurately
LSTM : Long-Short Term Memory
MLP : Multi Layer Perceptron
ConvBERT : Convolutional BERT
NER : Named Entity Recognizer

SYMBOLS

I	**:** Input Sentence
V	**:** Dependency Nodes
A	**:** Dependency Relation Matrix
morph	**:** The morphological features of a sentence
t	**:** Parts-of-speech tags of a sentence
Prop	**:** SRL tags of a sentence
D	**:** Inter-step Tree
C	**:** AMR Concepts
R	**:** AMR Relation Matrix
G	**:** AMR Graph
$\mathbf{L_c}$	**:** Concept Label set
$\mathbf{L_r}$	**:** Relation Label set
γ	**:** Add Edge Function
ζ	**:** Delete Edge Function
δ	**:** Add Node Function
φ	**:** Delete Node Function
ι	**:** Assign Label Function

LIST OF TABLES

LIST OF FIGURES

ABSTRACT MEANING REPRESENTATION OF TURKISH

SUMMARY

In this thesis, we focus on the Abstract Meaning Representation (AMR) for Turkish. The AMR is a sentence-level representation that summarises all semantic aspects of sentences. Its goal is to create representations that abstract from syntactic features. This is an attempt to group sentences with the same meaning in a semantic representation, regardless of the syntactic features of the sentences. It is also easily readable by humans, which is very convenient for researchers who want to conduct research in this area.

AMR is designed for the English language, but can be adapted to adapt to other languages by taking into account language-specific issues. To accomplish this task, it is mandatory to create an AMR guideline that defines language-specific annotation rules. In this thesis, we present Turkish AMR representations by creating an AMR annotation guideline for Turkish. Turkish is a morphologically rich, pro-drop and agglutinative language, which causes it to deviate from English AMR in its representations. In creating the Turkish guideline, we meticulously examine Turkish phenomena and propose solutions to define AMR representations for these deviant points. Besides, we present the first AMR corpus for Turkish that contains 700 AMR annotated sentences. Unfortunately, the creation of such resources is not an easy task, as it requires linguistic training and a large amount of time, and also requires a systematic annotation strategy. We adapt the model-annotate-model-annotate strategy to our annotation task, i.e., instead of dealing with all phenomena at once, we follow a stepwise path. First, we follow a data-driven approach and handle Turkish specific structures that are present in the data. In the second iteration, we use knowledge bases such as Turkish dictionaries and grammar books to cover all linguistic phenomena. This strategy allows us to build a corpus simultaneously. Instead of annotating the sentences from scratch, we use a semi-automatic annotation approach where a parser first processes the sentences and outputs the AMR graphs, which are then corrected/re-annotated by annotators (two native speakers). We implement a rule-based parser by inspiring the methods used in the literature. Our rule-based parser is very similar to the transition parsers, but its actions are driven by the rule list rather than an oracle. We design this parser in this way because our goal is to develop an unsupervised parser that utilizes the available sources. We evaluate our proposed solutions and the rule-based parser using the semantic match score (Smatch). This score shows the quality of our corpus and the accuracy of our parser. The inter-annotated agreement between our annotators is 0.89 Smatch score, the rule-based parser achieves a Smatch score of 0.60, which is a strong baseline for the Turkish AMR parsing task.

The final part of this paper deals with the development of a data-driven AMR parser. We formalize our parser as two steps containing a pipeline of multiple classifiers, each with different functionality. The first step of the data-driven parser is to identify concepts to be used in the AMR graphs. Nine separate classifiers are trained for this task. In addition, we add a post-processing step to capture the morphemes with predefined concept representations. In the second step, we attempt to build the AMR graph by extracting the minimum spanning graph from the fully connected graph obtained by connecting all the concepts identified in the first step. Our data-driven parsing system requires a prior alignment phase in which the training data is prepared. We present an aligner for morphologically rich and pro-drop languages. Our aligner strategy proposes an alignment algorithm in which first the concrete concept, then the abstract concept, and the morphology-derived concept are aligned using similarity and tree traversal, respectively. The proposed alignment algorithm achieves 0.87 F1 scores and outperforms aligners in the literature by a high margin, providing a relative error reduction of up to 76%. We evaluate the parsers introduced in this study on Little Prince sentences from the AMR corpus, whose gold tags are unfortunately not available. The external NLP tools are used for feature extraction, except for role labels. We develop a BERT -based end-to-end semantic role labeler that achieves an F1 score of 0.70 and outperforms the equivalent neural SRL system that employs bi-directional long-short-term- memory. Finally, we provide the Turkish AMR guideline, the Turkish AMR corpus, and the Turkish aligner on GitHub to support researchers explore Turkish AMR.

TÜRKÇENİN SOYUT ANLAM TEMSİLLERİ

ÖZET

Bu tezde, Türkçe Soyut Anlam Temsillerine (SAT) odaklanılmıştır. SAT, cümlelerin tüm anlamsal yönlerini bir araya getirerek cümle düzeyinde anlam temsilleri oluşturan bir anlamsal temsil formalizmidir. Bu temsillerin genel amacı, sözdizimsel özelliklerden soyutlanan temsiller yaratmak, bu sayede aynı anlama sahip cümleleri, tek bir anlamsal temsilde toplamamaktır. Ayrıca kolayca okunabilmesi, SAT temsillerini bu alanda araştırma yapmak isteyen araştırmacılar için oldukça uygun bir temsil haline getirmektedir.

SAT cümleleri köklü, yönlü ve çevrimsiz çizgeler olarak temsil eder. Cümle içerisindeki kelimeler bu çizgelerdeki düğümleri (*kavramlar*), kenarlar ise kelimeler ya da kelime grupları arasındaki anlamsal ilişkileri (*ilişkiler*) temsil eder. SAT cümlelerin sadece anlamı ile ilgilenir; cümlenin anlamına katkı sağlamayan kelimeler SAT çizgelerinden temsil edilmezler. SAT cümlelerin sözdizimsel özellikleri ile ilgilenmez, sadece anlamına odaklanır, bu da kelimelerin konuşma etiketlerinin önemini kaybetmesine neden olur.

SAT ilk olarak İngilizce için tasarlanmıştır, ancak dillere özgü yapıların çözümlenmesi ile diğer dillere uyarlanması mümkündür. Bu görevi gerçekleştirmek için dile özgü açıklama kurallarının tanımlandığı bir SAT kılavuzunun hazırlanması zorunludur. Bu tezde, Türkçe SAT açıklama kılavuzu hazırlanarak Türkçe SAT gösterimleri sunulmaktadır. Türkçe morfolojik açıdan zengin ve sondan eklemelibir dildir. Sahip olduğu yapım ve çekim ekleri, anlamsal temsillerinde İngilizce SAT'dan uzaklaşmasına neden olmaktadır. Türkçenin yapım ekleri yeni kelimeler üretirler ve bu kelimelerin anlamları üretilen kelime kökleri ile anlamsal olarak bağlantılı olabilir ya da tamamiyle farklı anlama sahip olabilir. Çekim ekleri ise karşımızı farklı görevler ile çıkabilir: (1) cümlelerin ögeleri arasındaki ilişkileri kurarlar (ör., ismin hal ekleri), (2) belirli bir göreve sahip olarak cümleye belirli bir anlam katarlar (ör., kip ekleri). Türkçeni bu yapısı SAT kurallarına göre ele alınmalıdır.Bu göz önüne alınarak, Türkçe kılavuzun hazırlanması sırasında Türkçe olguları titizlikle incelenmiş ve SAT temsilini tanımlamaya yönelik çözümler sunulmuştur. Yapılan çalışmalar ile SAT işaretleme yönergesi hazırlanmıştır.

SAT işaretleme yönergesi Türkçenin tüm dilsel olguları için SAT çözümünü içerecek şekilde hazırlanmıştır. Bu yönerge içerinde Türkçenin yapım ve çekim ekleri büyük yer tutar. İsimden yapım ekleri ile üretilen fiiller SAT çizgelerinde fiil çerçeveleri ile temsil edilirler. Eğer bu fiillerin fiil çerçeveleri yok ise, yeni çerçeveler üretilmedir; fakat bir istisna mevcuttür. Çok üretlen *-lA*, *-lAn*, *-lAş* bir çok isme eklenerek onları fiile dönüştürme kabiliyetine sahiptirler bu da bir çok fiil çerçevesinin üretilmesi

demektir. Bu durumdan sakınmak için yeni üretilen fiillerin Türkçe sözlükte olması, başka fiil çerçeveleri ile aynı anlamda ve edilgen çatılı fiil olmaması koşulları aranmıştır. Fiilden üretilen isimler bir aksiyon belirttiği için SAT'da eylem olarak temsil edilirler. Türkçenin sondan eklemeli yapısı sayesinde bu isimlerin kök durumlarına kolaylıkla ulaşılabilir ve bu kökler eylem olarak kullanılırlar. Türkçede yapım ya da çekim eki olmasından bağımsız olarak bir çok ek önceden tanımlı anlamlara sahiptir. Bu tip ekler SAT çizgelerinde kavramlar ile gösterilmelidir. Bu ekler SAT kurallarına uygun biçimde anlamınları karşılayacak en uygun kavramlar ile eşleştirilmiştir. Örneğin olumsuzluk yapım eki *-sIz* fiil çerçevesi olan 'yok.01', kip eki olan *-mAlI* yine fiil çerçeveleri 'öner.01', 'zorla.01' ve 'gerek.01' ile eşlenmiştir. Turkçenin İngilizceden ayrılan en önemli noktalarından biri de fiil çatılarıdır. Fiil çatıları cümlelerde özne ile yüklem arasındaki ilişkiyi değiştirdiğinden SAT çizgelerini doğrudan etkilemektedir. Çatı ekleri ile cümlelere verilen çatı anlamları için çalışmada özel kurallar konulmuştur. İşteş ve dönüşlü çatılı fiiller eylemi yapan ve eylemden etkilenen kişinin aynı olması sebebi ile fiil argumanları aynı yapılarak temsil edilmiştir. Ettirgen çatılı fiiller için ise 'yap.03' fiil çerçevesi üretilmiştir.

Bu çalışma kapsamında yanında, SAT açıklamalı cümlelerin sunulduğu Türkçe için ilk SAT derlemi de sunulmuştur. Malesef, bu tür kaynakları oluşturmak, dil eğitimi ve büyük miktarda zaman gerektirdiğinden kolay bir iş değildir ve ayrıca sistematik bir açıklama yaklaşımı gerektirir. Bu çalışmada, model-açıklama-model-açıklama stratejisi işaretleme sürecinde kullanılmıştır, başka bir deyişle tüm olguları tek seferde etiketlemek yerine aşamalı bir yol izlenmiştir. İlk aşamada, veri odaklı yaklaşımı izlenmiş ve üzerinde çalışılan veride bulunan Türkçeye özgü yapılar üzerinde durulmuştur. Isınma periodu adı verilen ilk etiketme aşamasında ana dili Türkçe olmayan dil bilimci ile çalışılmış, Küçük Prens kitabının ilk 100 cümlesi işaretlenmiştir. Isınma periyodundan sonra etiketleme hızını arttırmak amacı ile yarı otomatik etiketleme yaklaşıma başvurulmuş, işaretlemeye IMST derlemi üzerinden devam edilmiştir. Yarı otomatik etiketleme yaklaşımında kural bazlı bir ayrıştırıcı geliştirilmiştir. Sıfırdan cümleleri işaretlemek yerine, bir ayrıştırıcının önce cümleler üzerinde çalıştığı ve SAT grafiklerinin çıktısını aldığı, ardından işaretleyicilerin (iki ana dili Türkçe olan işaretleyici) bu çıktıları düzelttiği/yeniden işaretlediği yarı otomatik bir işaretleme yaklaşımı benimsenmiştir. Literatürde kullanılan yöntemlerden ilham alınarak kural tabanlı bir ayrıştırıcı geliştirilmiştir . Bu kural tabanlı ayrıştırıcı geçişli ayrıştıcılara çok benzese de, geçiş aksiyonları bir kahin tarafından değil, kural listesi tarafından yönetilmektedir. Buradaki temel amaç mevcut kaynakları kullanan denetimsiz bir ayrıştırıcı geliştirtir ve ayrıştırıcı bu doğrultuda geliştirilmiştir. Önerilen çözümler ve kural tabanlı ayrıştırıcı anlamsal eşleşme puanı (Smatch) ile değerlendirilmektedir. Bu skorlar, derlemin kalitesini ve ayrıştırıcı doğruluğunu gösterir. Kural tabanlı ayrıştırıcı, Türk SAT ayrıştırma görevi için güçlü bir temel olan 0.60 Smatch skoruna ulaşmıştır. İkinci aşamada ise, tüm dilsel olguları kapsayacak şekilde Türkçe sözlük, dilbilgisi kitapları gibi bilgi tabanlarından yararlanılmıştır.

SAT derlemi kalitesini ölçmek ve yarı otomatik işaretleme yaklaşımını değerlendirmek amacıyla deneyler dizayn edilmiştir. İlk deney kümesinde SAT derlemi üzerine yoğunlaşılmıştır. Derlemin genel doğruluk değerlendirmesi için işaretleyiciler arasındaki anlaşma skoru ölçülmüştür. İşaretleyiciler arasındaki anlaşma skoru 0.89

Smatch skorudur. Ayrıca hazırladığımız işaretleme yönergesinin anlaşılabilirliğini ölçmek amacıyla cümlelerdeki Türkçe dilsel olgular üzerindeki işaretleyici ortak anlaşma skoru ölçülmüştür. Cümlelerdeki dilsel olduguların temsil edildiği çizge parçaları üzerinden yapılan bu ölçüm sonucu 0.89 Smatch skorudur. Çalışmada kullanılan yarı otomatik işaretleme yaklaşımını iki farklı deney ile ölçülmüştür. İlk deney 1 işaretleyici 10 cümleden oluşan birbirine benzer iki kümeyi ayrı ayrı işaretlemiştir. Kümelerden bir tanesinin işaretlemesini sıfırdan yaparken ikincisinin işaretlemesini ise kural bazlı ayrıştırıcı üzeresinden yapmıştır. Bu iki farklı işaretleme yaklaşımında işaretleme süreleri ölçülmüş, önerilen yaklaşımın işaretleme süresini üç kat hızlandırdığı görülmüştür. İkinci deneyde ise 2 farklı işaretleyici aynı 25 cümleyi işaretlemiştir. Birinci işaretleyici ayrıştırıcı çıktıları üzerine işaretleme yaparken, iincisi sıfırdan işaretleme yapmış ve bu 25 cümlenin işaretleyici anlşama skoru ölçülmüştür. Bu sayede yarı otomatik işaretleme yaklaşımının işaretleyicileri yanlış işaretlemeye yönlendirip yönlendirmediği test edilmiştir. İki işaretleyicinin anlaşma skorları 0.85 olarak ölçülmüştür.

Bu tezin son kısmı, veriye dayalı bir SAT ayrıştırıcısının geliştirilmesine odaklanmaktadır. Veriye dayalı ayrıştırıcı, her biri farklı işlevselliğe sahip birkaç sınıflandırıcıdan oluşan ardışık yazılım zinciri içeren iki adım olarak biçimlendirilmiştir. Bu ayrıştırıcının ilk adımı, SAT grafiklerinde kullanılacak kavramları belirlemektir, bu kavramların bulunması için dokuz sınıflandırıcı eğitilmiştir. Ek olarak, önceden tanımlanmış kavram temsillerine sahip biçimbirimleri kapsayacak şekilde bir işlem sonrası adımı eklenmiştir. İkinci adımda ise, ilk adımda tanımlanan tüm kavramların birbirine bağlanmasıyla elde edilen tam bağlantılı grafikten minimum yayılma grafiği çıkararak SAT grafiği oluşturulmaya çalışılmıştır. Bu çalışmada sunulan veriye dayalı ayrıştırma sisteminde, eğitim verilerinin hazırlandığı ön hizalama aşamasını gerekmektedir. Literatürde İngilizce işin bir çok hizalayıc çalışması bulunmasına rağmen, morfolojik olarak zengin ve sözcük düşmesi içeren diller için bir hizalama çalışması yapılmamıştır. Bu tezde, morfolojik olarak zengin ve sözcük düşmesi içeren diller için bir hizalama yöntemi sunulmaktadır. Bu yöntem iki aşamadan oluşur. İlk aşamada çizge içerisindeki somut kavramlar sözcük vektörleri yardımı ile cümle içerisindeki kelimeler ike eşleştirilirler. İkinci adımda ise soyut ve morfolojiden türetilen kavramların hizalanmasına odaklanılır. SAT çizgesi üzerinde dolaşılarak komşuluk ilişkileri üzerinden cümle içerisinde sözcük karşılığı olmayan kavramlar hizalanır. Ayrıca önerilen hizalama yönteminde adlaşmış sıfatların hizalanması problemine de çözüm sunulmuştur. Yapısal benzerlik skoru kullanılarak adlaşmış sıfatlar uygun kavramlar ile hazlanmıştır. Önerilen hizalama algoritması 0.87 F1 skoru elde etmekte, literatürdeki hizalayıcılardan yüksek bir farkla daha iyi performans gösterip 76%'ya varan nispi hata azalması sağlamaktadır. Son olarak, bu tezde tanıtılan ayrıştırıcılar SAT derleminin Küçük Prens cümleleri üzerinde değerlendirilmektedir, ne yazık ki bu test kümesinin altın etiketleri mevcut değildir. Rol etiketleri dışındaki özellikleri çıkarmak için harici DDİ araçları kullanılır. Rol etiketleyici için ise sıfırdan ön-eğitimli dil modeli destekli bir görev çözümleyici geliştirilmiştir. Ön eğitimli dil modelleri görev çözümleme görevine uyarlanmış, parametreleri bu görev üzerinde ayarlanarak birer görev çözümleyiciye dönüştürülmüştür. Çözümleyi için ön eğitimli dil modelleri üzerinde testler yapılmıştır. BERT tabanlı çözümleyici 0.70 F1 puanları elde etmekte ve çift yönlü uzun-kısa süreli bellek kullanan eşdeğer sinirsel anlamsal

görev çözümleticide daha iyi performans göstermiştir.. Son olarak, araştırmacıları Türkçe SAT üzerinde çalışma konusunda desteklemek için Türkçe SAT kılavuzunu, Turkçe SAT külliyatını ve ayrıştırıcıyı GitHub'da paylaşıyoruz.

1. INTRODUCTION

The same expression may be verbalized differently, and yet they all refer to the same event. Figure 1.1 shows 5 different sentences with the same meaning but different syntax. In these sentences, there are two main events: 'desire' and 'believe', where the verb 'desire' can be realized as a noun 'desire' or as an adjective 'desirous'. As humans, we can interpret these sentences and conclude that they all mean the same thing. This is the ability that computers do not have.

```
The boy desires the girl to believe him.

The boy desires to be believed by the girl.

The boy has a desire to be believed by the girl.

The boy's desire is for the girl to believe him.

The boy is desirous of the girl believing him.
```

Figure 1.1 : Different realization of the two events 'desire' and 'believe' by the boy and the girl, respectively.

Researchers have been doing many researches on meaning representations (MRs) at different levels. FrameNet [5], VerbNet [6], and PropBank [7] are considered partial meaning representations and focus on the argument structure of verbal and nominal predicates. Some other MR frameworks represent meanings at the sentence level, sometimes even at the paragraph level. Discourse Representation Structures (DRS) [8], and Universal Conceptual Cognitive Annotation (UCCA) [1], Uniform Meaning Representation (UML) [9], and Abstract Meaning Representation (AMR) [10] are the most common, as they have extensive annotations. These frameworks have their own peculiarities: DRS is an intermediate-level meaning representation proposed in discourse representation theory [11], which is able to cross sentence boundaries. It is capable of representing various linguistic phenomena such as quantifiers, negation, and discourse structure, and can be translated into First-Order Logic, allowing the use

of existing inference engines. The Groningen Meaning Bank (GMB) [12] provides a large dataset annotated with DRS. UCCA presents a multilayered framework that aims to abstract from specific syntactic constructions to represent semantic relations. It represents sentences as directed acyclic graphs (DAGs) in which nodes are referred to as units. Units can be a terminal (i.e., a leaf node) or elements that are jointly considered as an entity according to certain semantic or cognitive viewpoints. The foundational treats sentences as a set of *Scenes* that describe movements or actions or temporally persistent states. Each scene contains a main relation; a secondary relation is also possible. It may also contain participants of the main relation. Figure 1.2 shows examples of UCCA annotation graphs.

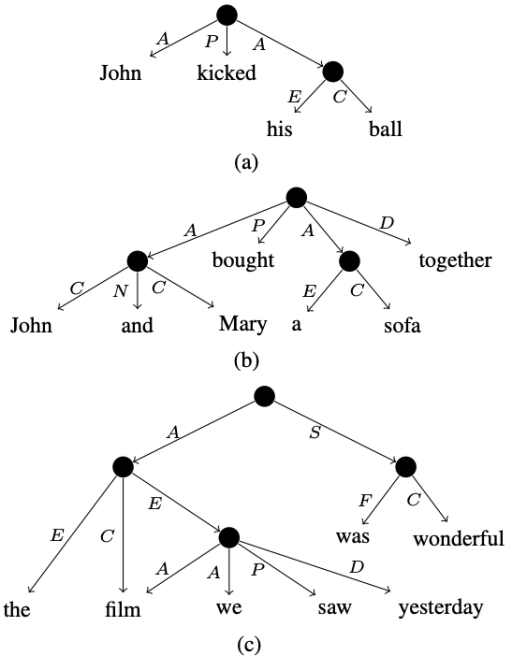

Figure 1.2 : UCCA representations of the sentences: “John kicked his ball” (a), “John and Mary bought a sofa together” (b), and ”the film we saw yesterday was wonderful (c) from top to bottom. This illustration is taken from [1]

AMR is a sentence level MR, which represents sentences as DAG and aims to abstract from syntax like UCCA. However, it diverges from UCCA by annotating senses, named entities, and relations. UMR is the extended version of AMR for other languages, especially morphologically complex, low-resource languages. The meaning represented in UMR goes beyond a sentence and sometimes extends to the

paragraph level, as it is able to capture linguistic phenomena such as coreference, temporal and modal dependencies. It can be shown in Figure 1.3.

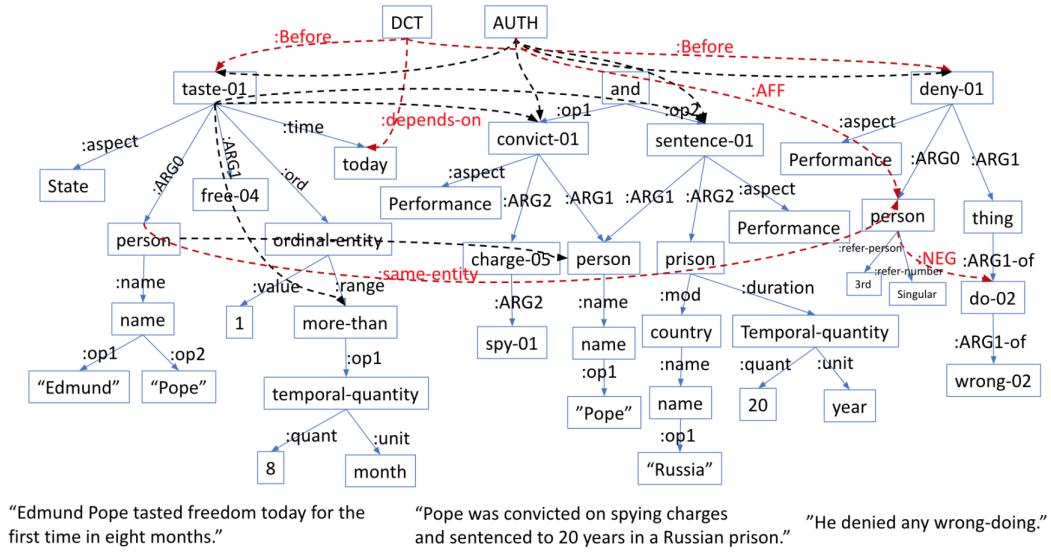

Figure 1.3 : An example of a UMR representation. This illustration is from [2]

1.1 Purpose of Thesis

Although there are many studies focusing on the syntactic features of Turkish and providing syntactically annotated data, the studies on the semantic features of Turkish are not yet mature enough. The previous efforts in this area mostly focus on partial meanings; unfortunately, there are no studies that investigate Turkish MR annotation at the sentence level. AMR is designed for English, however; there are several studies that show that it is possible to structurally match English AMR constructions with their counterparts in other languages by considering language-specific issues. Our main goals in this thesis are (1) to produce the first version of the Turkish AMR annotation guideline, (2) to produce the first Turkish AMR corpora, and (3) to develop an AMR parser for Turkish Building a gold standard for semantically annotated data and linguistic resources is a challenging task as it requires annotators with linguistic background and takes a lot of time. Our goal is to make use of existing resources and knowledge bases as much as possible in the annotation phase to speed up the whole annotation process. A semi-automatic annotation approach built on a parser developed based on manually extracted rules helps to speed up the annotation, which in turn leads

to a larger amount of data in a short time. The semi-automatic annotation is designed in such a way that first the parser outputs the AMR graphs and then the annotators correct erroneous parts of these outputs. After building an AMR-annotated corpus, our secondary goal is to provide a data-driven AMR parser for Turkish, to investigate the difficult points of Turkish AMR parsing, and to gain experience for our future AMR parsing studies.

1.2 Contributions of the Thesis

Our contributions in this thesis are as follows:

1. We present the first formal meaning representation for the Turkish language,
2. We present the first AMR representation framework for Turkish: the introduction of the AMR language-specific constructions of Turkish and the proposed AMR scheme, as well as an annotation guideline [1],
3. We first present the Turkish Abstract Meaning Representation Corpus, which contains 700 AMR annotated sentences,
4. We develop a rule-based Turkish AMR parser to accelerate the human annotation process using a semi-automatic approach (with a Smatch score of 60%),
5. We train the first data-driven AMR parsing system on the Turkish AMR Corpus that employ language-specific features,
6. We propose the first AMR aligner dealing with the alignment of concepts derived from morphemes for morphologically rich and pro-drop languages.
7. We language models to the Turkish SRL task for the first time.

[1] This work contains only the annotations differing from the English AMR guideline (`https://github.com/amrisi/amr-guidelines`). The extended version is available: `https://github.com/amr-turkish/turkish-amr-guidelines`

1.3 Organization of the Thesis

The thesis is organized as follows: Chapter 2 briefly explains the basics of AMR. Chapter 3 introduces the Turkish AMR representation framework by discussing the Turkish-specific constructions. Chapter 4 introduces the stages of corpus construction; our annotation methodology, the AMR annotation approach. In Chapter 5, we present the rule-based and the data-driven AMR parser, as well as the assistance tools used in the development of these parsers. Finally, Chapter 6 provides a brief summary and conclusion of the research as well as topics for future studies.

2. ABSTRACT MEANING REPRESENTATION

Abstract Meaning Representation is a framework developed mainly for English that represents meaning at the sentence level.

The main purpose of AMR is to provide a semantic representation that focuses only on sentence meaning and abstracts from syntactic idiosyncrasies. It also gathers balkanized semantic annotations (named entities, co-reference, semantic relations, etc.) into a single representation. Figure 2.1 shows that AMR is a knowledge-based meaning representation heavily relying on frame semantics (e.g., resources such as PropBank Frames or FrameNet) for linking predicate frames and entity knowledge bases such as DBpedia for linking named entity concepts[1]. AMR has its own definitions and annotation rules. In the following sections, we briefly explain the basics of AMR as a foundation for the topics discussed in the next chapter. For more information, see the AMR guidelines[2], which describes the details of AMR annotation in detail.

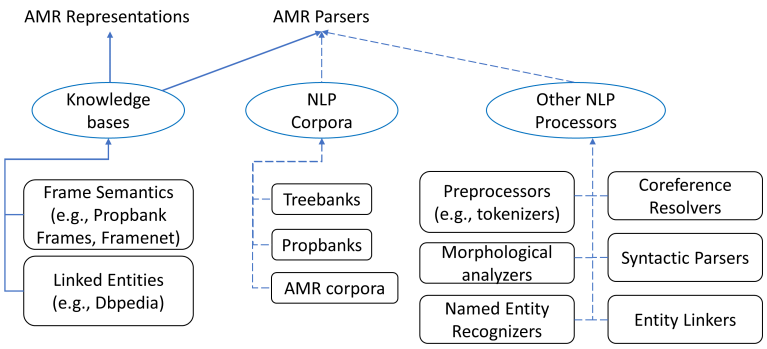

Figure 2.1 : AMR interaction with knowledge bases and other NLP resources. (Dashed lines represent optional interactions.)

2.1 AMR Fundamentals

AMR represents the meaning of a sentence in a single traversable DAG, in which nodes are called '*concepts*' and edges are called '*relations*'. Concepts represent elements

[1]AMR parsers often make use of additional NLP resources to construct the AMR structures from natural language sentences. These assistant tools are explained in Subsection 5.4.

[2]https://www.isi.edu/ ulf/amr/help/amr-guidelines.pdf

(i.e., words) of a sentence, and the semantic connection between these elements is denoted by relations. The AMR annotation depends heavily on The Proposition Bank (PropBank) [7], which contains verb frames of predicates along with their argument structure. AMR uses these frames to represent verbs and adopts their argument relations. Figure 2.2 shows an AMR representation of a sample sentence "The boy wants the girl to believe him" in graph notation. As can be seen in the figure, the nodes are labelled by concepts that carry the words in the sentences, and the edges show the semantic relationship between these concepts. In the example, there are two events: 'want' and 'believe', represented by their PropBank frames as 'want-01' and 'believe-01', respectively. The boy is *:ARG0* of the event want (wanter), the wanted thing (*:ARG1*) is the event believe. This event believe has an *:ARG0* (believer) who that is a girl, and its *:ARG1* is the same as the boy's *:ARG0* of the event want.

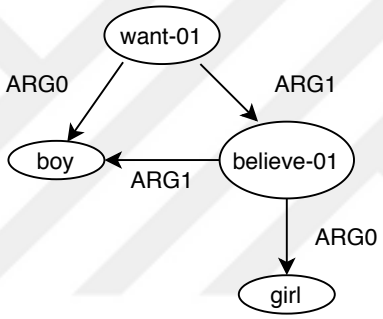

Figure 2.2 : An AMR representation of an example sentence "The boy wants the girl to believe him" in graph notation.

In AMR, each concept has a '*concept name*' and a '*variable*' that is used to instantiate the concepts so that the same concept can be used multiple times if needed (i.e., *reentrancy*). The variables are usually the initial letters of the concepts. They can be given a number if there are multiple concepts that start with the same letter. Figure 2.3 shows the same sentences as in Figure 2.2 in PENMAN notation [13][3]. As can be seen in the figure, variables and concept names are separated by a slash (/). Since the boy is related to both believe-01 and want-01, its variable 'b' is used to denote this concept in the second usage.

[3]The traditional logic format is also usable, but not common, as the other two are more flexible and easier for humans to read and write.

```
(w / want-01
   :ARG0 (b / boy)
   :ARG1 (b2 / believe-01
             :ARG0 (g / girl)
             :ARG1 b))
```

Figure 2.3 : An AMR representation of an example sentence "The boy wants the girl to believe him" in PENMAN notation.

There are two types of concepts in the AMR: lexical concepts and abstract concepts. While '*lexical concepts*' are the words that actually occur in the sentences, '*abstract concepts*' are predefined by the AMR and used for abstracting the meaning from the lexical words, such as names.

The AMR has about a hundred semantic relations, including frame arguments. Frame arguments are '*:ARGX*' roles and are inherited from PropBank. The other roles come from the AMR annotation and can be listed as non-core roles for general semantic relations (e.g., *:cause*, *:purpose*, etc.), roles for specifying date entities (e.g., *:weekday*, *:month*, etc.), roles of the form *:opx* used in conjunctions (e.g., *:op1*, *:op2*, etc.), roles of the form :prep-X used when no proper relation is available among the AMR relations. Most relations are variable, but there are also some relations that only take a constant value. *:polarity* is an example of such a relation, since it only takes a minus (-) to indicate negativity.

2.1.1 Abstracting away from syntax

The AMR strives for a more logical, less syntactical representation. To this end, it abstracts from syntactic features of sentences: Inflectional morphology for tense and number, voices, aspect, and word order. Moreover, there are no nouns and verbs in AMR. As a result of abstraction, we cannot recover the original sentences from the AMR graphs because multiple sentences may have the same AMR graph. For example, the following sentences have the same AMR graph depicted in Figure 2.4:

- The boy desires the girl to believe him.
- The boy desires to be believed by the girl.

- The boy has the desire to be believed by the girl.
- The boy's desire is for the girl to believe him.
- The boy is desirous of the girl believing him.

```
(d / desire-01
   :ARG0 (b / boy)
   :ARG1 (b2 / believe-01
             :ARG0 (g / girl)
             :ARG1 b))
```

Figure 2.4 : An AMR representation of the sentence "The boy desires the girl to believe him" in PENMAN notation.

2.1.2 Reentrancy

AMR allows a concept to participate in multiple relations, which is called '*reentrancy*'. Reentrancy is an important property of AMR because i) it transforms the tree structure of AMR representations into graphs. ii) it provides the ability to connect events when there is a common element. In Figure 2.2, the boy is *:ARG0* of 'want' and at the same time *:ARG1* of 'believe', which represents the reentrancy relation.

2.1.3 Inverse roles and reification

AMR deals with 'events' and captures the rough meaning of a sentence like "Who did what to whom, how, when, where, why?". Each AMR graph has a top root node, which is considered the focus of the sentence, and this node indicates the main event of the sentence. AMR also intends to extract sub-events from the constituents of the sentences, and the nodes that perform this role are placed closer to the root node than the nodes corresponding to other words in the constituents. Since predicates are the sentence elements that refer to the events, they take this role of focus and tend to be at higher levels in the AMR graph. However, there are cases where the focus may be on words other than predicates or on relations. This situation leads to the need for '*inverse roles*' and '*reification*'.

In AMR, when the focus is on words other than predicates, we can raise the corresponding nodes of such words to higher levels by inverse roles. In AMR, we can invert semantic roles by appending '*-of*' to their ends.[4] Inverse roles are also useful for obtaining a single root structure. Figure 2.5 shows an example of the inversion of '*:ARG0*'.

```
(b / boy
   :ARG0-of (s / sing-01)
   :source (c / college))
```

Figure 2.5 : An AMR representation of the sentence "There is a boy from the college who sang". Since 'boy' is emphasized, it becomes the root node and the inverse form of *:ARG0* is used.

AMR provides a solution for converting relationships into concepts. The transformation of a role into a concept is called reification. We can only reify non-core relations whose reifications are given in Table 2.1. As you may notice, reifications are tried to be mapped to available frames as much as possible. A new concept is defined only if there is no other predicate in PropBank that has the same meaning as the reification concept, and if '-91' is added to the end of the concept name. It should be noted that reification may lead to inconsistencies, since a single sentence may get two different AMR graphs: one with non-core relations, and the other with reifications. The correct use of reifications is still under discussion in the AMR community. For more information, we refer readers to the AMR guidelines[5].

2.1.4 AMR on different linguistic phenomena

AMR deals with language phenomena with different forms of representation as a consequence of dissociating from a syntactic structure. We outline the representation scheme for the main of these phenomena in the following subsections.

[4]The inverse role of *:domain* is *:mod*

[5]For a detailed explanation, see https://github.com/amrisi/amr-guidelines/blob/master/amr.md#reification

Table 2.1 : Non-core relations and their reification concepts in the AMR.

Relation	Reification	Domain	Range
:accompanier	accompany-01	:ARG0	:ARG1
:age	age-01	:ARG1	:ARG2
:beneficiary	benefit-01	:ARG0	:ARG1
:concession	have-concession-91	:ARG1	:ARG2
:condition	have-condition-91	:ARG1	:ARG2
:degree	have-degree-92	:ARG1	:ARG2
:destination	be-destined-for-91	:ARG1	:ARG2
:duration	last-01	:ARG1	:ARG2
:example	exemplify-01	:ARG0	:ARG1
:extent	have-extent-91	:ARG1	:ARG2
:frequency	have-frequency-91	:ARG1	:ARG2
:instrument	have-instrument-91	:ARG1	:ARG2
:li	have-li-91	:ARG1	:ARG2
:location	be-located-at-91	:ARG1	:ARG2
:manner	have-manner-91	:ARG1	:ARG2
:mod	have-mod-91	:ARG1	:ARG2
:name	have-name-91	:ARG1	:ARG2
:ord	have-ord-91	:ARG1	:ARG2
:part	have-part-91	:ARG1	:ARG2
:polarity	have-polarity-91	:ARG1	:ARG2
:poss	own-01, have-03	:ARG0	:ARG1
:purpose	have-purpose-91	:ARG1	:ARG2
:quant	have-quant-91	:ARG1	:ARG2
:source	be-from-91	:ARG1	:ARG2
:subevent	have-subevent-91	:ARG1	:ARG2
:time	be-temporally-at-91	:ARG1	:ARG2
:topic	concern-02	:ARG0	:ARG1
:value	have-value-91	:ARG1	:ARG2

2.1.4.1 Nominals that invoke predicates

The AMR is interested in the events, regardless of the parts of speech, and the verbal form of the nominals that invoke predicates is represented in the AMR graphs. The verbalization process may be done by using OntoNotes [14] predicates, whose use is highly supported by AMR. However, there are some cases where a nominal has a substantially different meaning than its verbal form. In such cases, AMR retains the meaning of the nominal and represents it as it is. While 'destruction' has the

same AMR representation as destroy-01, AMR does not decompose the meaning of 'president' into preside-01 and person.

2.1.4.2 Modals and copulas

The AMR does not have a complicated way of representing modality. It ignores grammatical tenses and auxiliary verbs such as 'would'[6] and expresses syntactic modals simply by predicate frames shown in the Table 2.2. As you may notice, a verb frame can be assigned to more than one modal, depending on its meaning in the sentence.

Table 2.2 : The modals and their corresponding AMR concepts.

Modal	Concept
can	possible-01
may	possible-01, permit-01
might	possible-01
would rather*	prefer-01
should	recommend-01

Copulas are closely related to grammatical structures (i.e., tenses), so the AMR abstracts the representation of copulas as much as possible. The relation '*:domain*' is used to represent sentences saying "Noun is noun." instead of the verb frame of '*be*'.

2.1.4.3 Named entities and wikification

AMR represents named entities with the abstract concepts (i.e., the concepts that specify the entity category) followed by a role and a concept, both with the same name: '*name*'. There are several named entity categories, including organizations, places, facilities, events, products, etc. Even pets, ships, and computers are treated as named entities in AMR. Also, AMR forces us to choose the most specific applicable category as the named entity. If there is no defined entity type that applies to the entity, 'Thing' should be used. In many cases, different names refer to a single named entity, e.g. US, U.S., United States, or United States of America. AMR unifies these different uses of the same object and represents them in a canonical form. To this end,

[6]The exceptional case is marked with an asterisk (*) in the Table 2.2

it references related English Wikipedia entries and labels named entities with a '*:wiki*' role. Figure 2.6 shows an example of the AMR representation of named entities.

```
(a / award
   :wiki "Nobel_Prize"
   :name (n / name
            :op1 "Nobel"
            :op2 "Prize"))
```

Figure 2.6 : The AMR representation of 'the Nobel Price'

2.1.4.4 Questions

AMR has a reserved concept '*amr-unknown*' to indicate question forms. AMR graphs of questions are constructed as if they were simple statements. Then amr-unknown is placed where the concepts/graph fragments corresponding to the answers to the questions are found. The question "What does Lokum eat?" *:ARG1* of eat.01, which can be shown in Figure 2.7, accommodates amr-unknown. The same approach is used to represent yes/no questions, but in such cases, the relation *:polarity* is used with amr-unknown.

```
(e / eat.01
       :ARG0 (p / pet
              :wiki -
              :name (n / name :op1 "Lokum")))
       :ARG1 (a / amr-unknown)
```

Figure 2.7 : The AMR representation of 'What does Lokum eat?'

2.2 Evaluation

AMR uses the Smatch Score (similarity match score) [3], a metric that calculates the degree of overlap between two AMR graphs. The relations encoded in the AMR graph can also be represented as triples. Each AMR triple has one of the following forms: relation(variable, concept), relation(variable1, variable2). The evaluation metric measures the precision, recall, and f-score of the triples in the second AMR

compared to the triples in the first AMR based on the number of propositional overlaps (number of common triples). First, the variable names in both AMR graphs should match. The difficulty is that the variable names may not match between the two AMRs. To calculate the propositional overlap based on different variables, the Smatch score is calculated for each one-to-one variable match between the two AMRs, and the highest F1 score is accepted as the similarity score of the two AMR graphs. For example, consider the variables in AMR graph1=x,y,z and the variables in AMR graph2=a,b,c. Figure 2.8 shows the calculation of the Smatch score for the example.

```
                    M     P     R     F
x=a, y=b, z=c:      4   4/5   4/6  0.73
x=a, y=c, z=b:      1   1/5   1/6  0.18
x=b, y=a, z=c:      0   0/5   0/6  0.00
x=b, y=c, z=a:      0   0/5   0/6  0.00
x=c, y=a, z=b:      0   0/5   0/6  0.00
x=c, y=b, z=a:      2   2/5   2/6  0.36
---------------------------------------
smatch score:                      0.73
```

Figure 2.8 : Calculation of the Smatch Score between AMR Graph1 with variables x,y,z and AMR Graph2 with variables a,b,c. This illustration is from [3].

Since the alignment problem is NP -complete, finding the optimal alignment takes a long time even if the AMR graphs are small. Integer linear programming and Hill-climbing are used to solve this problem.

3. ABSTRACT MEANING REPRESENTATIN OF TURKISH

Turkish is a strong representative of MRLs. Its complex morphology and agglutinative nature pose several challenges in creating AMR representation of sentences, although AMR ignores syntactic features of sentences. The first and essential step in creating an AMR representation for Turkish is to develop a guideline addressing the challenges of Turkish-specific constructions, which requires redefining several linguistic phenomena (e.g., voices, nominal verbs, etc.) in terms of AMR. In this chapter, the Turkish AMR framework is presented, including our proposed solutions for the challenging Turkish-specific constructions in terms of AMR. First, AMR studies on different languages that have similar problems to Turkish are presented, then the Turkish AMR framework is introduced. Section 3.2 provides the phenomena that need to be adapted to Turkish. It should be noted that The use capital letters in the representation of suffixes is a tradition in Turkish NLP to depict the possible phonological changes under different circumstances (vowel/consonant harmony rules); A denotes 'a' or 'e', H: 'ı', 'i', 'u', or 'ü', and I: "ı" or "i", C: 'c' or 'ç'. In this representation, a parenthesis used around a letter means that the use of that letter may be omitted under different phonological occurrences. e.g., the suffix -(I)ş may be seen as -ş, -ış, or -iş. This chapter is based on our previous paper [15], where we investigated the utility of AMR for the Turkish language.

3.1 Related Work

AMR has attracted the attention of many researchers in the NLP community [16,17] and has been used for various applications, including summarization [18]–[20], text generation [21]–[30], machine translation (MT) [31], sentence compression [32], event extraction [33,34], human-robot interaction [35], and natural language understanding in dialogue systems [35,36].

Although AMR was developed for English, adaptation to other languages is an open research topic in the AMR community. Chinese is the language with the most

AMR studies after English. [37] has presented the Chinese AMR framework, also called CAMR, which addresses the differences between English and Chinese. The Chinese AMR studies deal with headless relative constructions. [37,38], reduplication [38], elliptical constructions [39]. The Chinese AMR corpus [38] is the largest among non-English languages with about 10K sentences, including the annotation of the Chinese translation of the novel "The Little Prince" and the CTB Chinese Treebank [40]. [41] shows the points where Spanish AMR annotations differ from English. In their work, the authors discuss the phenomena of null subject, possessive pronoun, reciprocal and reflexive voices. In addition, these studies also paved the way for Portuguese AMR. [42,43] also provide solutions for Portuguese specific structures. [44,45] introduced an annotation specification for Korean [46] in which copula structures for a morphologically rich language were studied in detail. Other studies reporting on AMR adaptation to other languages include [47] for Czech and [48] for Vietnamese. The corpora released by these studies are very modest in size (ranging between 50 to 1.5K sentences) compared to English and Chinese. Moreover, [49] proposes a modified version of AMR. In the study, PropBank arguments with predefined roles (e.g., *actor* instead of *:ARG0*) are mapped to proper argument relations, which can overcome some PropBank-related problems such as fine-grained sense disambiguation and high startup costs. The positive impact of this modification on annotation times and parsing accuracy is shown in the paper. Last but not least, [50] presents a cross-language semantic representation as a simplified version of AMR. The presented scheme expresses only essential semantic features and other important features of a sentence, such as predicate roles and linguistic relations.

3.2 Turkish AMR Framework

There are a variety of suffixes in Turkish. Each suffix has different functions, including forming new words by derivation and establishing relationships between sentence components. Derivational suffixes (DSs) are responsible for forming new words whose meaning may be inferrable by base words, or may be completely different. [51] states that according to the Turkish Language Association, there are 759 root verbs, 2380 verbs derived from nouns, and 2944 verbs derived from other root verbs via

derivational suffixes. The ability of these suffixes to create new words may require additional attention at the concept creation stage of AMR, since a derived word may be represented by multiple concepts. On the other hand, inflectional suffixes (ISs) are ignored in AMR because they have some aspects of their grammatical functions, such as plurality and tense. However, ISs require special treatment in Turkish because (1) they can play the role of establishing relational links between the constituents of a sentence, (2) they can have predefined meanings that create concepts (i.e. personal markers, modality markers, etc.). In addition, a word may have multiple suffixes that have different predefined meanings or functions. This may result in a word forming a complex AMR graph rather than a single concept. For example, the word 'görüştürüldü' (with separated morphemes as 'gör-üş-tür-ül-dü' meaning *S/he is made to have an interview with someone.*) has correspondingly 1 derivational, 2 voice (causative and passive) and 1 inflectional morphemes (past tense 3rd person singular) and its AMR representation is quite complex. Regardless of the type, the suffixes that establish relations between the constituents of a sentence should be mapped to the correct AMR relations, and the others that derive concepts must be mapped to concepts according to their meaning. This mapping could be straightforward in some cases, but the majority of suffixes cannot be mapped directly due to Turkish-specific constructions, which requires a revision of the AMR framework.

We aim to create an AMR framework for Turkish. Our first priority is to make our annotations parallel to the English AMR framework. For this purpose, we adopt annotations defined for the phenomena that do not require additional processing in Turkish, such as the representation of conjunctions, all kinds of numbers, named entities, etc. The rest of this chapter is reserved for the phenomena where problems arise that need to be dealt with. In the following subsections, we first describe the phenomena and then present our solution.

3.2.1 Verbal derivation from nominals

The derivation of verbs from nominals (nominal verbs) is a phenomenon frequently used to form verbs in Turkish. There are several suffixes for the verbal derivation of nominals. The highly productive suffixes *-lA*, *-lAş*, *-lAn* differ from the others by their

high verb production capacity. These suffixes are capable of converting a large number of nominals into verbs, some of which may not in the dictionary and are dynamically derived in daily usage. An example of this is 'eflatun**laş**' (*to take lilac- colour*), which is not in the Turkish dictionary but is one of the words used in daily speech. For simplicity, we will use the abbreviations HPSs for these highly productive suffixes and HPVs for nominal verbs derived with them in the following.

Turkish has an immense number of nominal verbs, so it is a difficult task to cover them all. [51] claims that creating a nominal bank to which nominal verbs are linked is a viable approach to solving the framing problem. However, in their follow-up study [52], they present a different solution in which the most frequent nominal verbs are included in the Turkish PropBank. We believe that framing nominal verbs considering the frequency of their observation is essential, as there may be some verbs whose meaning changes over the years.[1]. The authors of [52] also tried to solve the problem of dynamic derivation of HPVs by defining frames with x-roots: xlA, xlAş, xlAn. Although this seems to be a suitable solution to incorporate such high-productivity structures into PropBank, it is not suitable for AMR due to the shortcomings listed below:

- x-rooted frames treats all HPVs as the same, even though there may be differences between their argument structures, even if they appear to be grammatically the same
- AMR is interested in events, which should be represented as root nodes of graphs and subgraphs. x-rooted frames violate this convention
- AMR aims at a representation that is easily readable by humans, x-rooted frames make it difficult to read.

Given these considerations, we believe that a different approach should be taken for Turkish AMR. As we mentioned earlier, creating new frames for the nominal verbs is an appropriate solution. For the representation of nominal verbs in general (except HPVs), we use the existing PropBank frames if available. Otherwise, we create/propose new frames for them. FootnoteFor framing, we follow the previous

[1] We strongly recommend applying this solution to verbs derived from verbs as well

efforts for Turkish [51], and we create the new frames using the predicate framing editor provided in [53] according to the PropBank framing guidelines [54]. To represent HPVs, one approach is to add new frames for the verbs that deviate from the x-rooted frames, similar to [51]. However, this approach (1) complicates the verb framing process, as it is left to annotators to decide which verbs deviate from the x-rooted frames; (2) is prone to framing errors, as new frames can be created for the verbs that need to be represented by the x-rooted frame. Moreover, HPVs should be represented with expressive frames that eliminate the above drawbacks. The other approach is to create new verb frames for all HPVs, which exponentially increases the number of verb frames and leads to an unmanageable PropBank.

Our solution for representing HPVs is to create a new frame for an HPV if it satisfies all of the following conditions simultaneously:

1. The verb should be present in the Turkish dictionary,
2. One should not be able to represent the verb with another verb frame from PropBank,
3. One should not be able to represent the verb as a passive form of another verb frame from PropBank.

Productive derivational suffixes regularly have certain meanings [55] and add these meanings to noun lemmas when deriving HPVs. We use the adjective ”literally” to express the most basic meaning of these new verbs after attaching an HPS (with a predefined meaning). Some HPVs may acquire other meanings over time and become completely new verbs, while their literal meaning is rarely used in everyday life. For example, when the HPS *-lAn* is attached to ’ev’(*home*), the verb ’ev**len**’ is created and its meaning becomes ”to marry” instead of ”to buy a house”. The main reason we expect a verb to be present in the Turkish dictionary to create a new frame (the 1st condition above) is that the dictionary lists all these (additional or main) meanings. This listing does not include dynamically derived HPVs and helps us to determine them. Also, we want to avoid creating too many frames and prefer to use existing frames rather than creating new ones. The 2nd and 3rd conditions above ensure that

if there is a frame that has the same meaning as the HPV, the existing frame is used, regardless of whether the HPV is passive or active. Figure 3.1 shows three such HPVs. The first example 'gueneş**len**' (*sun*) is included in the dictionary, but its frame is not in the Turkish PropBank.[2]. It also has a special meaning that is not to be taken literally. We create a new frame for gueneşlen. "get a sun". The rest of this section illustrates our solution for the suffix *-lAn*, but this solution also applies to the other two suffixes *-lA* and *-lAş*.

```
(g / güneşlen.00
     :ARG0 (o / o)
     :time (g2 / gün
             :time (d / date-entity
                     :season (s / sonbahar))
             :mod (g3 /güzel)))
```

Figure 3.1 : The AMR representation of "Güzel bir sonbahar gününde güneşleniyordu."

The HPS *-lAn* generally adds the meaning of getting the thing/state expressed by the noun. It can dynamically derive verbs from almost any noun (e.g., buz**lan**, where 'buz' means ice in English and this verb is used as "to become icy"). Although these verbs are grammatically correct, they may not be included in the dictionary. For example, someone might say "Arabalandım." (*I got a car.*), where the noun 'araba' (*car*) is changed to 'araba**lan**' (*get a car*) to express that he/she has bought a new car in everyday speech. As we mentioned before, it is not possible to create a new frame for all derived HPVs. Therefore, we solve this problem by considering the meanings of such HPVs. When a derived verb carries its literal meaning, we use an appropriate PropBank frame that matches the derived verb. In the case of *-lAn*, when the HPV means "to have the object represented by the nominal", it should be assigned to the frame 'ol.4' (*to get*) (associated with the nominal concept) rather than creating a new frame. Figure 3.2 shows the example. Similarly, if it means "to become the state of the nominal" (e.g., 'hüzünlenmek' (*become sad*)), it should be assigned to frame 'ol.2'

[2]It should be noted that missing predicate frames are represented by adding a -00 tag to AMR predicate concepts as a suggestion for later inclusion in the knowledge base

(*become*), depicted in Figure 3.3, instead of creating a new frame, even though the verb exists in the dictionary (due to the violation of the 2nd condition above).

```
(o / ol.04
     :ARG0 (p / person
              :name (n / name
                       :op1 “Ahmet”))
     :ARG1 (a / araba)
```

Figure 3.2 : An AMR representation of the nominal verb ‘arabalanmak’ in the sentence “Ahmet arabalandı” (*Ahmet got a car*).

```
(o / ol.02
     :ARG0 (b / ben)
     :ARG1 (h / hüzün)
     :time (g / gör.01
              :ARG0 b
              :ARG1 (r / resim
                       :mod (e / eski))))
```

Figure 3.3 : An AMR representation of the nominal verb ‘hüzünlenmek’ in the sentence “Eski resimleri görünce hüzünlendi”(*S/he became sad when s/he saw old pictures*) .

Another feature of *-lAn* is the conversion of nominals into passive verbs such as ’yasaklan’ (*be banished*) or ’kurulan’ (*be dried*). AMR ignores the voice structure and therefore leads to redundant concept creation when new verb frames are created for such HPVs (3rd condition above). However, it should be carefully decided whether the verb is in the passive form. Some verbs derived with *-lAn* can be used as both active and passive verbs. For example, the verb ’av**lan**’ (*hunt*) in the sentence “Fareler kedi tarafından avlandı.” (*The mice were hunted by the cat*) is passive, while it is active in “Dişi kurt tek başına avlandı.” (*The she-wolf hunted alone.*).

3.2.2 Verbal nominalization

Nominals that invoke predicates (verbal nominals) give a sense of action to a clause without a predicate. From the following sentence “I wonder about the baby’s smile.” we can understand that the baby performs the smiling actions. The noun smile denotes the event of which *:ARG0* is the baby. As described in section 2.1.4.1, they should be

represented with frames. However, representing such nominals in semantic annotations requires additional resources for English. While semantic role labeling systems use the Nominal Bank (NomBank) [56], which provides frames for such nouns, the English AMR uses sense-tagged verbs from OntoNotes [14].

Turkish also has nouns that differ in derivation from English. Several types of nominals (i.e., nouns, adjectives, adverbs) can be created dynamically from verbs by using suffixes. The main difference is that stems provide a direct link between verbs and nominalized verbs. This feature allows linking these nominals directly to their associated verb frames in PropBank, without the need for additional resources as in the case of English. Table 3.1 shows examples of verbal nominals in different types and their associated verb frames.

Table 3.1 : Noun, adjective and adverb samples

Type	Example	Verb Frame
Noun	Kız **geleceğini** söyledi. the girl - **that she will come** - said *The girl said that she will come.*	gel.01 (come.01)
Adjective	**Konuşan** kişi benim babam. **who is speaking** - the person - my father *The person who is speaking is my father.*	konuş.01 (speak.01)
Adverb	**Sorulunca** cevap ver. **when you are asked** - answer *Answer when you are asked.*	sor.01 (ask.01)

Figure 3.4 illustrates an example of a verbal nominal. The nominalized verb 'söylediğimi' (*what I said*) is formed by adding the subordinating suffix *-dIK* and 3rd person possessive suffix *-im* to the verb stem 'söyle'. We can easily link this nominal to the verb stem 'söyle.01' (*say.01*). Although linking nominals to their associated verb frames seems straightforward, there are some phenomena (e.g., adverbial subordination and headless relative constructions) that require special treatment. In the following sections, we describe these and present our solutions.

```
(u / unut.01
        :ARG0 (o / o)
        :ARG1 (t / thing
                :ARG1-of (s / söyle.01
                        :ARG0 (b / ben)))
```

Figure 3.4 : An AMR representation of the sentence 'Söylediğimi unutmuş.' (*S/he forgot what I said*).

3.2.2.1 Non-finite adverbial subordination

Turkish has a vast number of finite and non-finite adverbial clauses. The most commonly, used adverbial clauses are the non-finite ones, which contain subordinate verb forms [55]. The subordinated verb forms are called converbs and constructed by converbial suffixes. These suffixes transform verbs into adverbs by adding predefined meanings that help establish relationships between sentence constituents. We map these suffixes to the corresponding AMR relations. We should also keep in mind that Turkish suffixes may have different meanings depending on the context; this is also true for converbial suffixes. Therefore, during annotation, they may carry a different meaning than the one we assigned to them. In such situations, annotators should assign these suffixes to the correct relations accordingly. Table 3.2 shows the assignment of some suffixes to AMR relations.

As shown in the table, we define new relations *:prep-while* and *:prep-after*, since the existing AMR relations do not cover the meaning of these suffixes. It should be noted that due to the fact that all these relations in English AMR come from prepositions, they are defined as 'prep-X' convention. However, in Turkish, they are rather post-positions. In order to make our annotation parallel to English, we follow the prep-X convention. However, we believe that these prefixes present syntactic problems and should be removed in a universal scheme.

3.2.2.2 Headless relative constructions

Headless relative constructions are a type of relative clauses whose head nouns are not used explicitly, but are implicitly inferred by the readers with the help of the context.

Table 3.2 : Some suffixes forming converbs and their corresponding AMR relations.

Suffixes	Relation	Example
-Ip	:prep-after	Arabaya bin**ip** gitti. to the car - by getting into - went *S/he got into the car and left.*
-ArAk	:prep-by	Ağla**yarak** yanımıza geldi. crying - next to us - s/he came *S/he came to us crying.*
-mAdAn	:prep-without	Bugün kahvaltı yap**madan** okula gitti. Today - breakfast - without doing- to school - went *S/he went to school today without having breakfast.*
-dHkçA	:prep-as	Tren hızlan**dıkça** ağaçlar sıklaşmaya başladı. train - as moved faster - trees - denser - got *As the train speeded up, the trees were getting denser.*
-HncA	:time	Eve gid**ince** beni ara. to home - when you go - me - call *Call me when you go home.*
-kAn	:prep-while	Telefonla konuşur**ken** kapı çaldı. with the phone - while speaking - the door - rang *The door rang while I was speaking on the phone.*

Turkish is a pro-drop language; its pro-drop nature allows the omission of object or subject pronouns that may occur with nominalized verbs. The main noun (i.e., the head), which may or may not be alive, can be dropped. Whether it is omitted or not, it should be represented in the AMR annotations. We add concepts for the omitted heads; the added concepts can be either person or thing, depending on the context. Figure 3.5 shows an example where the added concept is 'person', since the reading action can be performed by a human.

```
(k / kazan.01
        :ARG0 (p / person
                :ARG0-of (o / oku.01
                        :ARG1 (k2 / kitap
                                :quant(ç / çok)))
        :ARG1 (y / yarış.01))
```

Figure 3.5 : An AMR representation of the sentence "Çok okuyanlar yarışmayı kazandı." (*Those who read a lot of books won the competition.*).

3.2.3 Nominal derivation from nominals

The representation of nominals derived from nominals could be complicated. The DSs used for their derivation may correspond to new concepts/relations or give rise to new nominals, which may be completely different words used in the AMR representation as the concept of the root word.

The nominals that have new meanings are new concepts since they are independent of the root nominals. They acquire exceptional meanings which is not easily deducible from the root's meaning, and appear in the dictionary as separate lemmas. The noun 'güney' (*south*) is an example of this type of nominal. It is derived from the noun g'ün' (day) and the semantic connection between 'gün' and 'güney' is not obvious. Moreover, 'güney' is a different word in the dictionary and should be presented in the AMR with an independent concept.

DSs corresponding to new concepts/relations have one or more predetermined literal meanings. The derived nominals carry this meaning and can be easily related to the root nominal. Such DSs belong to the highly productive suffix class, and the nominals derived from them may not be included in the dictionary. The suffixes *-CA*, *-lI*, *-sIz* are standing examples of such DSs and have several meanings. The suffix *-CA* has an interesting characteristic. When attached to nominals, it gives rise to many different meanings represented by the relations *:manner*, *:quant*, and *:duration*. When attached to pronouns, the word expresses a person's point of view and is considered an independent event. Therefore, we use the predicate 'düşün.01' (think) to represent this case. Figure 3.6 illustrates an example of the annotation of the suffix *-CA*.

```
(d / düşün.01
      :ARG0 (b / ben)
      :ARG1 (m / mümkün.01 :polarity -
            :ARG1 (g / geç.01
                  :ARG0 (b2 / biz)
                  :ARG3 (k / köprü
                        :mod (b3 / bu)))))
```

Figure 3.6 : An AMR representation of the sentence "Bence bu köprüden geçemeyiz." (*I think we can't cross this bridge*).

There are cases where two of the above scenarios for the same suffix occur in different contexts. The suffix *-sIz* can have the meaning that (1) the nominal described lacks what is expressed by the root (e.g., 'para**sız**' (*pennyless*)) (2) the non-participation in an event of what is expressed by the root (e.g., 'sensiz' (*without you*)). In these uses, the suffix adds the meaning of negativity to the root nominals, the representation of which is discussed in the subsection 3.2.15. On the other hand, the suffix loses its predefined meaning in the produced nominal and becomes part of the nominal, as in 'telsiz' (*walkie-talkie*), where the literal meaning would be 'without wire'. The produced nominal should represent the word as an independent concept.

3.2.4 Modality

In Turkish, modality suffixes are used to express modalities such as possibility, obligation, and permission. As described in Section 2.1.4.2, predicate frames are used to represent syntactic modalities in English AMR. For Turkish, we map Turkish modality suffixes to some selected predicates without changing the sentence meaning in parallel with the English AMR. Since Turkish PropBank does not provide a frame for the sense of possibility, contrarily to English PropBank, we createthe verb frame 'mümkün.01' (*possible*) for this sense. The frame has only one argument (i.e. *:ARG1*) indicating the possible event. Furthermore, modality markers can have more than one sense, which complicates frame mapping difficult due to fact that the meaning of the marker must first be distinguished. Furthermore, a verb may have more than one modality suffix at a time, each of which must be separately mapped to predicate and represented in the AMR. Table 3.3 shows some common modality suffixes with their corresponding verb frames. As you can see, the markers *-Abil* and *-mAlI* may have different meanings depending on the context. Note that modality markers are not only a tool to express the meanings provided by modals, there are also some nominals that have the same meaning as modality markers. To make the annotation consistent, we assign these nominals to the same frames as modality markers.

Table 3.3 : Modality samples.

Modality Sense	Turkish Suffix	Verb Frame	Example
Possibility	-Abil,	mümkün.01	Çekilişi kazan**abil**irim. (I may won the lottery.)
Permission	-Abil	ver.09	Polis, "gid**ebil**irsiniz", dedi. (Policeman said: "you can go".)
Necessity	-mAlI	gerek.01	Enfekte olmamak için maske tak**malı**sın. (You must wear mask not to get infected.)
Obligation	-mAlI	zorla.01	Ödevimi yap**malı**yım. (I have to do my homework.)
Recommendation	-mAlI	öner.01	Bu filmi kesinlikle izle**meli**sin. (You should definitely watch this movie.)

3.2.5 Copula

In Turkish, there are many forms of copula, including zero- copula, be copula, evidential copula, conditional copula, etc. We follow the English AMR guidelines and represent copula markers with the relation :domain. However, this relation does not fit the meaning of some nominals with copula markers and the conditional copula, so an additional solution should be found. The representation of the conditional copula is straightforward, since AMR has a non-core relation *:condition* to represent the conditional meaning. We simply use this relation. Some nominals may take the position *-dA* as well as become complementary nominals in light verb constructions. At the same time, they can be inflected by the copula marker, which cannot be represented by the relation *:domain* and becomes the predicate of the sentence. The nouns 'yaşın**da**' (*at the age of*), 'farkın**da**' (*be aware of*) and 'güven**de**' (*be safe*) are the examples of such nominals, and can be used with the light verb 'ol' (*become*). To inline with the English AMR, we use the reification frame of the relation *:age* which is 'yaşlan.01' (*to get older*), and the AMR graph is depicted in the Figure 3.7. It should be noted that the frame 'yaşlan.01' in Turkish PropBank does not have *:ARG2*; it should be updated to the version that has the same argument structure as its English counterpart (i.e., age-01). The following examples gives the example AMR representations of the nominal mentioned.

```
(y / yaşlan.01
        :ARG0 (b / ben)
        :ARG2 (t / temporal-quantity
                :unit (y / yıl)
                :quant 10)
```

Figure 3.7 : An AMR representation of the sentence “On yaşındayım.” (*I am ten years old.*).

For the other nominals such as ‘güvende’ and ‘farkında’, we propose to represent them by using ‘ol.02’ (to take the state denoted by a noun or adjective) if their meaning matches with the nominals, otherwise we propose to create a new verb frame for these nominals. The example Figures 3.8 and 3.9 show AMR representations of ‘güvende’ and ‘farkında’, respectively.

```
(o / ol.02
        :ARG0 (b / ben)
        :ARG1 (g / güven)
        :location (e / ev
                :poss b))
```

Figure 3.8 : An AMR representation of the sentence “Evde güvendeyim.” (*I am safe at my home*).

```
(o / ol.27
        :ARG0 (b / ben)
        :ARG1 (t / thing
                :ARG1 (y / yap.01
                        :ARG0 (s / sen)))
```

Figure 3.9 : An AMR representation of the sentence “Yaptıklarının farkındayım.” (*I am aware of what you did.*).

3.2.6 Voices

Voices describe the relationships between the predicate and the subject; therefore, voice markers (VSs) have the ability to change the argument structure of verbs [55], affecting their AMR representations.

Turkish has four voice structures: reciprocal, reflexive, causative, and passive. Turkish PropBank does not provide verb frames for VS -inflected verbs and treats them with their stems and additional features. However, there are two possible solutions for their representation: (1) creating verb frames for all verbs, which ends up with numerous verbs frames and raises the same problem we described in the previous section 3.2.1. Our general approach to frame representation is to avoid creating redundant frames. (2) Following [52], additional arguments are generated, leading to compatibility issues between English and Turkish AMR. We believe that an AMR-friendly approach should be adopted to represent the voices of Turkish to avoid emerging AMR incompatibilities. It should be noted that the passive voice does not change the argument structure of verbs, because in this structure the performers of the events are unknown. We handle this by leaving the argument*:ARG0* empty, as in English. In the following paragraphs includes our solutions for representing voice structures in focus.

Reciprocal verbs express actions performed together or against each other and are formed with the reciprocal suffix *-(I)ş*. Only a few transitive and intransitive verb stems could be transformed into reciprocal verbs. [52] represent such verbs by favouring the number of agents, which is not a suitable approach for representing verbs indicating reciprocal involvement in the action. Our solution is to make the agents performing the action reciprocally *:ARG0* and *:ARG1* of the verb. Figure 3.10 shows the AMR representation of the sentence "İki çocuk birbirleriyle güreşiyor." (*Two children wrestling with each other*). The subject 'iki çocuk' (*two children*) is both *:ARG0* and *:ARG1* of the reciprocal verb 'güreş' (*wrestling*).

```
(g / güreş.01
      :ARG0 (ç / çocuk
            :quant 2)
      :ARG1 ç)
```

Figure 3.10 : An AMR representation of the sentence "İki çocuk birbirleriyle güreşiyor." (*Two children wrestling with each other*).

Reflexive verbs denote actions that affect the person performing the action, either directly or indirectly (e.g, 'yıkanmak' *to wash oneself- to take a bath*, 'giyinmek'

(*to wear oneself to get a dress*, etc.). The reflexive voice marker *-(I)n* is attached only to transitive verbs, transforming them into reflexive verbs. To represent their argument structure, [52] proposes to define a new semantic role 'A01', since PropBank conventions do not allow annotating an argument with multiple roles, which is allowed in AMR. The same approach proposed for reciprocal verbs is followed for these verbs. We intentionally do not distinguish between reciprocal and reflexive voice structure in AMR, since the person performing the action and the person affected by the action are identical. The only difference is that reciprocal verbs include multiple actors. Moreover, AMR aims to abstract from reflexives, co-references, etc. [10]. It is worth to say that an alternative to our current solution could be to use some specific pronouns (e.g., 'birbiri' (*each other*) for reciprocity and 'kendi' (*oneself*) for reflexivity) to distinguish these two structures. Figure 3.11 illustrates the AMR representation of the sentence "Dün yıkandım." (*I took a bath yesterday.*) As shown in the figure, *:ARG0* and *:ARG1* of the verb 'yıka.01' (*wash*) are the pronoun 'ben' (*I*).

```
(y / yıka.01
      :ARG0 (b / ben)
      :ARG1 b
      :time (d / dün))
```

Figure 3.11 : An AMR representation of the sentence "Dün yıkandım." (*I took a bath yesterday*).

The causative structures are formed by the causative suffixes (*-dIr*, *-t*, *-It*, *-Ir*, *-Ar*, *-Art*), which can be attached to transitive or intransitive verbs [55], e.g. 'boyatmak' (*to make somebody paint samething*), 'getirtmek' (*to make somebody bring something*), etc. [52] define a new role 'ArgA' to indicate the causer of an action, which seems to be a suitable solution to incorporate the verb framing of the causative structure. As we mentioned above, the addition of new roles removes the parallelism between the English and Turkish AMR, since the English AMR does not need an additional role to represent the causative voice formed by the predicate 'make'. The arguments of the predicate are the main action performed (*:ARG1*) and the agent causing the action (*:ARG0*). To parallelize the Turkish AMR with the English AMR, we use the verb 'yap.03', which is the newly created as an equivalent for make-02 in the English

AMR. Figure 3.12 shows the AMR representation of the causative verb ‘kestir’ (make

```
(y / yap.03
      :ARG0 (b / ben)
      :ARG1 (k / kes.01
            :ARG0 (a / anne
                  :poss b)
            :ARG1 (s / saç
                  :poss b)))
```

Figure 3.12 : An AMR representation of the sentence “Saçımı anneme kestirdim.” (*I had my mother cut my hair.*).

somebody cut). It should be pointed that Turkish allows for nested voice structures. Such structures should be represented considering their context. For the sentence “Terziye elbisemi diktirttim.” (I made my tailor to make somebody to sew my dress.), two nested ‘yap.03’ predicates should be used in the AMR annotation.

3.2.7 Case markers

Case markers describe relationships between constituents of a sentence, one of which (locative case) may give different meanings to sentences depending on the context. One should decide which relationship to use depending on the sentence meaning.

3.2.8 Personal markers and possession

In Turkish, the explicit use of the subject is not mandatory. The doer of an event is referred by the personal suffixes at the end of the predicates. In the sentence “Kedimi seviyorum” (*I love my cat.*), the suffix “*-m*” (the last letter of the verb indicating I), for example, indicates who loves the cat. This way of using the subject is called “null subject”. The subject revealed by the personal suffixes is accepted as the of the event and is referred to in AMR by nominative pronouns. Since they stand for the subject, they are connected to the predicates that carry the meaning of the event with the related argument, which is usually *:ARG0*. It should be noted that the absence of a person marker in the predicate refers to the 3rd person singular subject.

A situation similar to the null-subject phenomenon in verbs also occurs in nominals with possessives. Turkish is a pro-drop language; possessivity is expressed by

possessive suffixes attached to nominals and/or the possessor. The use of the possessive pronoun is optional, as it can be dropped in the case of a null subject. The possessive person is also revealed by the possessive suffixes. By applying the same solution to the case of the null subject, we represent the omitted pronoun as an AMR concept. We then relate this concept to the possessed nominal using the relation '*:poss*'. Figure 3.13 shows the AMR representation of the sentence used in the previous example. While the personal suffix *-m* reveals who is performing the action (i.e., to love), the owner of the cat is revealed by the possessive marker *-m*.

```
(s / sev.01
        :ARG0 (b / ben)
        :ARG1 (k / kedi
                :poss b))
```

Figure 3.13 : An AMR representation of the sentence "Kedimi seviyorum." (*I love my cat.*).

3.2.9 Noun compounds

The first noun of bare noun compounds may indicate the gender, profession or nationality of the second noun (e.g., 'Türk doktor' (*Turkish doctor*), 'kadın sanatçı' (*female artist*), etc.), which is indicated by the relation :mod. Moreover, the first noun can also specify the material of the second noun (e.g., 'çelik kapı' (*steel door*), etc.), we use the relation :consist-of for such compounds. Some of the bare noun compounds can be the names of the cooked dishes (e.g., 'şiş kebap' (*shish kebab*), etc.), which are treated as named entities in AMR. Adjective-noun compounds consist of an adjective and a noun, majority of which are written as single words (e.g., 'karabiber' (*pepper*), 'akciğer' (lung), etc.), therefore we accept them as single words and represent them with concepts. The last type of compounds are *-(s)I* compounds and there are no specific semantic relations between them. The first word may specify the source of the second noun (e.g., 'inek sütü' (*cow's milk*) or indicate the purpose of the second noun (e.g., 'çay bahçesi' (*tea garden*)). We suggest to select the most appropriate relation among the non-core AMR relations to represent this type of compound. Note that Turkish allows stacked noun compounds, which may produce nested representations. An example of such a representation is given in Figure 3.14.

```
(f / fiyat
      :poss (k / kılıf
            :purpose (r / raket
                  :purpose (t / tenis))))
```

Figure 3.14 : An AMR representation of the sentence ‘tenis raketi kılıfı fiyatları’ (*tennis racket cover prices*).

3.2.10 Turkish clitics

Clitics are particles that form phonological units but not morphological units in an independent word [55]. In Turkish, there are several clitics, some of which have a predefined function (e.g., ‘mI’ transforms sentences into their interrogative form and its representation is described in the Subsection 3.2.16). The others might have different meanings for a particle in the language due to the productive nature of Turkish. In this section, we examine the clitics *-dA* and *-ki* and discuss the annotation of the different meanings for each case.

The clitic *-dA* is a conjunction and a connective with several functions: additive, adversative, enumerative, and continuative. The first three have AMR equivalence, while the last has none, since it does not contribute to sentence meaning. The meaning contribution of *-dA* is as follows: (1) Its additive meaning is also similar to its English counterpart (i.e., too) and is represented by *:mod*. (2) It can be used to express the meaning of ”but”. We represent this meaning by using the verb frame ‘karşılaştır.01’ (*compare.01*). The unstressed *-ki* mean ‘so that’. In such sentences, *-ki* is represented with the relation *:purpose*.

3.2.11 Postpositions

A postposition is a word that indicates the relationship of a noun, pronoun, or clause to another word in a sentence. Turkish has a large number of postpositions (Turkish has no prepositions [55]), some of which are called prepositions in English. Similar to English prepositions, we map postpositions to either verb frames or relations. This mapping is done taking into account their meaning and selecting the AMR

components/verb frames that best fit their description. Table 3.4 gives some pairs of postpositions and AMR components.

Table 3.4 : The concept mapping of some postpositions.

Postpositions	AMR components
için	:purpose
ile	:instrument
A göre	söyle.01
A kadar	:time
A rağmen	:concession
dAn beri	:time

There are several postpositions that differ from each other and have multiple meanings (e.g., 'gibi' (*like*), 'böyle' (*like this*), 'şöyle' (*like that*)). There is no predefined mapping for such words, they should be mapped to the correct AMR component depending on their meaning in context. Figure 3.15 and Figure 3.16 show two sentences with the postposition 'gibi', each with a different meaning.

```
(b / benze.01
        :ARG0 (s / sevgili
                :poss (b / ben)
                :mod (e / eski))
        :ARG1 (s / sen))
```

Figure 3.15 : An AMR representation of the sentence "Eski sevgilim senin gibiydi." (*My ex was like you.)*.

```
(ş / şirket
   :example (a / and
        :op1 (c2 / company :wiki "IBM"
                :name (n / name :op1 "IBM"))
         :op2 (c3 / company :wiki "Google"
                :name (n2 / name :op1 "Google"))))
```

Figure 3.16 : An AMR representation of the sentence "IBM, Google gibi şirketler" (*Companies like IBM, Google*).

3.2.12 Phrasal verbs

Light verb constructions or phrasal verbs in Turkish are formed by a nominal plus a light verb such as 'ol-' (*bebecome*), 'et-' (*make/do*), 'al-' (*get/take*), etc. [51] examines the light verb constructions according to the sense of the nominals: (i) a light verb can contribute to the meaning comparatively easily, (ii) a light verb does not contribute to the overall meaning. The authors have treated the latter category as a new predicate since the meaning conveyed by the nominal (e.g., 'teşekkür et-' (*to thank*)). In the AMR case, regardless of whether the nominal has a strong sense, we use the PropBank senses from the reference frame of the light verb.

3.2.13 Reduplications

There are 3 types of reduplications in Turkish: (1) Emphatic reduplications emphasise the quality of an adjective and adverb. For this type of reduplication, we use the :degree relation. In Figure 3.17, the word 'yapayalnız' is an example of emphatic reduplication and is derived from the adjective 'yalnız' (*alone*). We represent this word with *:degree (ç / çok)* to reflect the emphasis of emphatic reduplication on the representation.

```
(y / yaşa.01
   :ARG0 (b / ben)
   :manner (y / yalnız
           :degree (ç / çok)))
```

Figure 3.17 : Emphatic reduplication: An AMR representation of the sentence "Yapayalnız yaşıyorum." (*I live all alone.*).

(2) The m-reduplication involves the repetition of a word or phrase in modified form; the word is followed by the second word, which is exactly the same word, only the initial letter changes and becomes *m*. The m-reduplication is used to expand the range of the first word, and the m- part gives a similar meaning represented by the verb frame 'benze.01' (*resemble.01*) depicted in Figure 3.18.

```
(o / oku.01
   :ARG0 (b / biz)
   :ARG1 (a / and
      :op1 (k / kitap)
      :op2 (ş / şey
            :ARG0-of (b2 / benze.01
                  :ARG1 k))))
```

Figure 3.18 : m-reduplication: An AMR representation of the sentence "Kitap mitap okuduk." (*We read books and stuff.*).

(3) Doubling reduplication is similar to English and is formed by the simple repetition of a word, which may be a noun, an adverb, or an adjective. We also show its effects on meaning using the same solution we proposed for emphatic reduplication depicted in Figure 3.19.

```
(y / yağ.00)
   :ARG0 (k / kar)
   :manner (i / ince
          :degree (ç / çok))
```

Figure 3.19 : Doubling reduplication: An AMR representation of the sentence "İnce ince bir kar yağıyordu." (*it is snowing flaky.*).

3.2.14 Quantifiers and scope

AMR does not have a deep representation for quantifiers. It only canonicalizes their position. If a quantifier has a negative meaning, it is treated as a negative word and represented with *:polarity*. It should be noted that AMR does not advise on the placement of negation with respect to quantifiers, and we prefer to keep it on the predicate. Moreover, the sentences containing both a word meaning 'all' and a negativity marker *-mA* require the second negativity on the quantifier. The need for the second negativity is explained using the example in Figure 3.20. The figure shows an AMR representation of the sentence "Oğlanların hepsi gitmedi." (*Not all of the boys left.*). The example tells us that some of the boys went, but not all of them. There are some boys who did not go. In the AMR representation, we put the second negativity to show that the boys who did not go, which means "not all the boys".

```
(g / git.01
        :ARG0 (o / oğlan
                :mod (h / hepsi
                        :polarity -))
        :polarity -)
```

Figure 3.20 : The AMR representation of the sentence “Oğlanların hepsi gitmedi.” (*Not all of the boys left.*)

3.2.15 Negation

In Turkish, there are several ways to negate a statement, listed below.

1. The negativity suffix ‘*-mA*’ attached to verbs is frequently used to negate a sentence.
2. The negation of the be copula is expressed with ‘değil’.
3. The verb ‘*-sIz*’ (*non-existent*) gives the sense of negation to the sentence.
4. The very productive suffix ’*-sIz*’ may denote non-involvement in an event described by the nouns to which it is attached.
5. Turkish also has negative affixes, similar to English.

In general, we use the relation ‘*:polarity*’ to represent negation according to the English AMR guideline. However, the situations described in items 3 and 4 above cannot be represented with this relation, a concept with a negative sense would be more appropriate. Turkish PropBank provides a verb frame ‘yok.01’ with the meaning ‘do not exist’. We use this frame to represent the negative meaning of ‘yok’ (item 3) and ’*-sIz*’ (item 4), and an additional *:polarity* is not needed. Figure 3.21 shows our solution for representing the very prolific ending ‘*-sIz*’.

```
(a / adam
        :ARG0-of (y / yok.01
                :ARG1 (ş / şapka)))
```

Figure 3.21 : An AMR representation of the sentence ‘şapka**sız** adam’ (*man without hat*).

3.2.16 Questions

AMR uses amr-unknown as a concept explained in the Subsection 2.1.4.4, we apply this convention to interrogatives in Turkish. One difference between the Turkish and English AMR annotation refers to the position of 'mI' in the sentence. It transforms the sentences into interrogatives where the immediately preceding words or word groups are questioned and the questioned part becomes the focus. Therefore, we change the focus and placement of the amr-unkown. Figures 3.22 and 3.23 show the change of focus depending on the position of 'mI'.

```
(g / git.01
        :ARG0 (s / sen)
        :ARG4 (e / ev)
        :polarity (a /amr-unknown))
```

Figure 3.22 : An AMR representation of the sentence "Eve gittin **mi**?" (*Did you go home?*).

```
(e / ev
      :ARG4-of (g / git.01
                  :ARG0 (s / sen))
      :polarity (a / amr-unknown))
```

Figure 3.23 : An AMR representation of the sentence "Eve **mi** gittin?" (*Was home where you went?*).

In the first AMR annotation, the questioned part is whether the subject went home or not, so the focus is on the action itself. The second shows the same sentence, where only the position of mI changes. In this sentence, the focus shifts to 'eve' (*to home*), which comes immediately before the clitic mI. Thus, we make 'ev' (*home*) the root node.

4. TURKISH AMR CORPUS

In this chapter, we use the Turkish AMR guideline (3) introduced in the previous chapter to create an AMR annotated corpus. First, we explain our annotation methodology and the steps we went through during the annotation process. Then, we give the statistics of our corpus and present the evaluation results for the annotation quality of the resource in terms of complete sentences and phenomenon fragments. This chapter is based on our previous paper [15], where we investigated the resource creation of AMR for the Turkish language.

4.1 Annotation Methodology

Developing an annotation guideline is a challenging task and taking into account different linguistic phenomena increases its complication. However, designing an annotation specification is an essential step in building language resources. Specifications need to be updated frequently as data is annotated, since the information to be added to the specifications (e.g., exceptions) and the way it is expressed may arise during annotation.

In our annotations, we follow a similar annotation cycle as 'MAMA', depicted in Figure 4.1,(model-annotate-model-annotate) [57] and adopt the backbone of the [58] annotation model whose content we define.

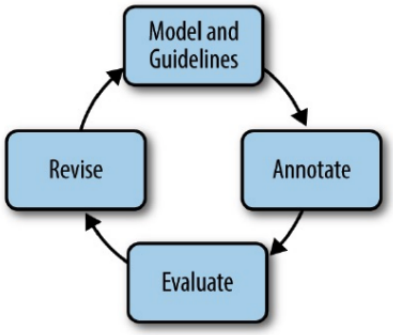

Figure 4.1 : The MAMA Sub-cycle, taken from [4]

Our modeling, depicted in Figure 4.2, proceeds as follows:

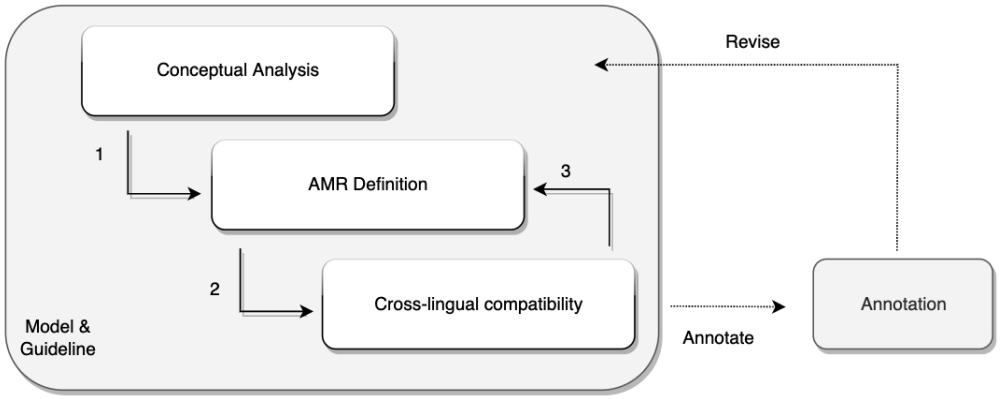

Figure 4.2 : The annotation cycle that we followed during AMR annotation

1. **Conceptual Analysis:** It is used to explore a phenomenon and its properties.
2. **AMR Definition:** An AMR definition of a phenomenon is made and its representation scheme is formed.
3. **Cross-lingual compatibility:** The compatibility of the created representation with the English AMR is checked.

The definition of an AMR specification (hereafter referred to as specs) for a phenomenon begins with the conceptual analysis phase. In this phase, we explore the phenomenon and aim to decide whether the phenomenon needs to be adapted to Turkish. The conceptual analysis is followed by the AMR definition phase, in which the specs for the phenomena are created.

The representation can be converted from English without additional processes such as named entities, quantities, etc., or it can be created from scratch for the phenomena for which it was decided in the previous phase that adaptation is needed (e.g., voice structures). Either produced or transformed, the compatibility of a created AMR representation with English is checked in the cross-linguistic compatibility phase. This phase prevents possible representation errors. If there is an incompatibility between the newly created representation and its English counterpart, the process returns to the previous step, otherwise the specs are accepted and manual annotation starts. During

annotation, one may encounter cases that cannot be covered by the specs. In such cases, the modeling is restarted and the specs are revised.

4.2 Annotation Tool

A web-based editor developed by [59] has been used for AMR annotation. The text commands and graphical buttons provided by the editor are used to create AMR annotations. In addition, this editor has a dictionary that includes a search engine that allows you to check the previously annotated phrases with a complete list of examples, which is an important source to ensure the consistency of the annotation. Figure 4.3 shows the user interface of the annotation tool. The annotation starts with the 'top' button, then other buttons help build the AMR graph in PENMAN notation. The tool automatically assigns variables to the concepts entered, so there is no need to enter variables for the concepts, which prevents possible erroneous annotations (i.e., multiple concepts with the same variable).

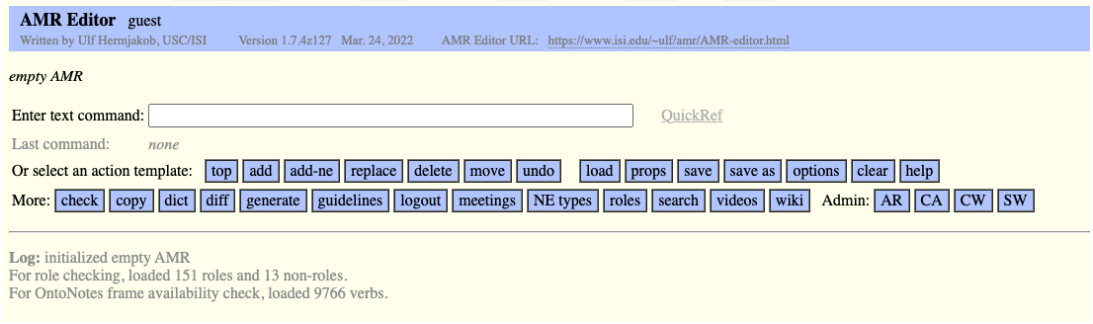

Figure 4.3 : AMR annotation editor for English

We have used an updated version of the AMR annotation tool that allows us to enter non-English characters. Apart from this advantage, the updated version has no connection to with the resources (e.g., the search engine, the example list). Annotators who want to see these features need to check them through the original tool, which may take additional time.

4.3 The Annotation Turkish Corpus

In our annotations, we have been through a cycle explained in section 4.1, involving two iterations to obtain the final framework and corpus.

The first iteration starts with a warm-up period. We start by annotating the first 100 sentences from the novel 'The Little Prince' (tLP) by Antoine de Saint-Exupery. During the annotation process, we worked with a foreign linguist who had experience with AMR and, especially in the modeling phase of our annotation cycle, conducted a preliminary investigation of Turkish phenomena. The preliminary investigation of Turkish structures and a warm-up phase of annotation, which we will discuss below [60]. The reason we started with tLP is that this novel has been used in several studies for different languages [10,38,42], which allows us to compare AMR on the same text between different languages. At the end of the warm-up period, we obtained (1) the first draft of the Turkish AMR specifications containing the annotation results and (2) 100 AMR annotated sentences.[1]

After the warm-up period, we continued the annotation using the semi-automatic annotation approach, which consists of two consecutive steps: parsing and human correction. Due to our limited human annotation resources, we preferred to follow this approach from scratch, as annotation from scratch is quite a time-consuming process that requires knowledge of PropBank structure and in-depth analysis of sentence meaning. Adapting earlier resources such as TreeBank and PropBanks could speed up the annotation process as they provide lower level linguistic annotations (i.e. morphology, dependencies, PropBank annotations) as a gold standard.For the Turkish language, there is such a resource, the 'Turkish Propbank' [51], which is based on the 'IMST Turkish Treebank' [61]. We used these resources with a semi-automatic approach. To this end, we developed a rule-based tree-to-graph parser (presented in section 5.2) whose backbone depended on the first drafts of the specifications produced at the end of the warm-up phase. Two native language annotators worked on the output graphs to produce the final output, rather than annotating from scratch. During the semi-automatic annotation, the specifications continued to be updated using a data-driven approach guided by (i) the sections of the Turkish grammar books on the grammatical phenomena occurring in the data in question and (ii) the English AMR

[1]The annotation was performed by the linguist and one of the annotators simultaneously.The inter-annotator agreement between the annotators was measured as 92% in terms of the Smatch score.

guideline. By the end of this first iteration, we had produced the first version of the specifications and the Turkish AMR corpus of 700 sentences.

The second iteration is dedicated to generalizing our specifications to cover all Turkish-specific phenomena, including those not present in the data we focus on. During the first iteration, we realized that the data we used alone was not sufficient to finalize the specifications and that we needed additional knowledge-based resources. Therefore, we expanded our research to include a variety of knowledge-based sources: Turkish grammar books, the Turkish dictionary. In addition, AMR studies in other languages were also examined. As a result of this iteration, the Turkish AMR annotation system presented in the 3 chapter was developed and the corpus was re-annotated accordingly by an annotator. This iteration also helped to generalize the specifications as various samples were collected outside the corpus for inclusion in the Turkish AMR guideline.

4.4 The Corpus

The Turkish AMR corpus contains 700 sentences, of which 100 sentences are from tLP and 600 from IMST. IMST has text gathered from 8 genres [62] (e.g., news, novels, interviews, etc.). The average sentence length of the 600 IMST sentences in our corpus is 11 tokens, of which 16% (i.e., 99 sentences) consist of less than 5 words. It should be noted that sentence length is not a reliable measure to make a conclusion about sentence complexity, since a short Turkish sentence may be very complex in terms of AMR (e.g., "Aradığımı buldum sandım." (*I thought I found the thing that I was looking for.*). Therefore, the proportion of complex sentences in the 600 IMST sentences is investigated. A complex sentence is defined as a sentence that contains at least one subordinate clause in addition to the main clause [55]. Such sentences make up a larger part of IMST sentences. 60% of them (357 sentences) have a complex structure. The sentences from tLP are relatively simpler than the IMST sentences, where the average sentence token is 8 and 36% (i.e., 36 sentences) contain at least one subordinate clause.

In the AMR corpus, there are 6655 concepts (5849 from IMST, 806 from tLP) and 7131 relations (6277 from IMST, 854 from tLP). The detailed statistics can be found in the Table 4.1. During the annotation process, we updated the Turkish PropBank by either

creating new predicate frames or adding new arguments to existing ones. We created 14 predicates in total, 7 of them for idiomatic expressions. We updated 2 predicates (i.e., 'yaşlan.01' and 'yap.03'). We believe that this shows that our proposed solution is reasonable and does not lead to a high number of new predicate frame generations.

Table 4.1 : The detailed statistics of the Turkish AMR Corpus.

	IMST	tLP
Abstract Concepts	79	18
Reification Concepts	82	8
Named Entity Concepts	207	5
Concepts from Headless Rel. Cons.	6	3
Concepts from Voice Cons.	62	2
Concepts from Modalities	67	13
Concepts from Reduplication Cons.	13	4
PropBank Arguments	2559	377
Non-core Relations	3948	363
Inverse Relations	363	40
:prep-X	73	5
Constants	188	21

4.5 Evaluation

To evaluate our corpus, we measure the inter-annotator agreement (IAA) between our annotators.We randomly selected 100 sentences from IMST at the end of the second iteration of our MAMA cycle and a second annotator re-annotated them. We evaluate the quality of the corpus by computing two different Smatch scores. One is computed based on an AMR graph as traditional to measure the IAA of the entire sentence's annotation. The other is computed on the AMR graph fragment concerning only one linguistic phenomenon to understand the clarity of our guideline. 90% of the selected sentences (i.e., 90 sentences) have at least one linguistic phenomenon and the IAA over this set is 0.89, which is equal to the IAA of the whole corpus. Table 4.2 shows the IAA of each phenomenon described in chapter 3 in the columns where 'full' and 'fragment' represent the traditional and phenomenon-based scores, respectively. We excluded personal markers from the second evaluation because personal markers are obligatory in Turkish. The results show that our annotators agreed on most AMR

annotations of phenomena with a smatch score higher than 80%. 'Reduplication' and 'Verbal Derivation from Nominals' have scores below 80%. We observed that these two were incorrectly annotated by one of our annotators. However, the sample sizes (1 and 3 sentences) are too small to draw any conclusions.

Table 4.2 : IAA on the AMR graph fragment concerning only one linguistic phenomenon.

Phenomena	# of sentences	full	fragment
Verbal Derivation from Nominals	3	0.75	0.77
Noun that invoke predicates	49	0.89	0.91
Non-finite Adverbial Subordination	11	0.93	0.86
Headless Relative Constructions	8	0.96	1
Null Subject	46	0.90	0.98
Modality	12	0.85	0.90
Voices	9	0.91	0.86
Nominal Derivation From Nominals	10	0.84	0.87
Pronoun-dropping	14	0.92	0.97
Reduplication	1	0.68	0
Copula	12	0.86	0.92

5. AMR PARSER OF TURKISH

This chapter introduces two AMR parsers for Turkish: a rule-based AMR parser and a data-driven parser. Both are developed following the guidelines presented in chapter 3 and evaluated using the corpus described in chapter 4. First, we present the rule-based parser (in section 5.2) used as an assistant tool in the creation of the corpus. We then introduce our data-driven parser (in Section 5.3), which consists of several decision trees connected in series. We also introduce an aligner (in Section 5.4.1), first developed in the literature for morphologically rich and pro-drop languages. Section 5.4 is reserved for the assistant tool developed to build parsers. This chapter is based on our previous papers [15,63,64], where we investigated the utility of AMR parsing for the Turkish language.

5.1 Related Work

5.1.1 AMR parsing

AMR parsing is an automatic process of converting sentences into their AMR representations. Recently, parsing studies have attracted the attention of many researchers in the natural language processing (NLP) community, and there have been two shared tasks at CoNLL 2019 and 2020 [65,66]. There are four major approaches for parsing AMR, including (i) Graph-based approaches, (ii) Transition based approaches, (iii) Seq2seq based approaches, and (iv) sequence-to-graph (seq2graph) approaches. Graph-based approaches [67]–[73] first identify concepts in sentences and then construct the possible edges using graph-based algorithms. Transition-based parsing [74]–[84] uses a series of actions to process a sentence and generate an AMR graph by either inserting a new node or adding a new edge. Seq2seq-based approaches use sequence-to-sequence models for AMR parsing by linearizing AMR graphs [85]–[91]. As a final approach, sequence-to-graph approaches build AMR graphs incrementally, such that the models jointly predict new nodes with their links

at each time step [92,93]. Although many AMR parsing studies continue to focus on English, there are also significant efforts for non-English languages. [94,95] presents a multilingual AMR parser that adapts a transition-based English AMR parser trained on automatically annotated data for Italian, Spanish, German, and Chinese. [89] uses several transfer learning techniques for multilingual AMR parsing. Brazilian Portuguese is another language in which AMR parsing studies are actively continuing. [96] presents a rule-based parser.

5.1.2 Assistant tools

5.1.2.1 AMR alignment

JAMR [67] aligner is the first AMR aligner in the literature. It uses heuristic rules and searches the matching cases greedily. In this method, the aligner proceeds downward from the first rule and searches for a fuzzy match based on the rule currently being processed. While some rules are applied to all nodes by traversing the entire graph for each rule, others are applied only to specific nodes (e.g., entity names). The TAMR [79] aligner is an extension of the method presented in JAMR with a focus on meaning. It adds syntactic and semantic matching to the list of JAMR rules, using semantic and morpho-semantic matching together with fuzzy matching. It uses the morphological meaning database [97] to establish connections between verb-invoking nouns and their verb frames (e.g., example - exemplify). [98] uses statistical machine translation enriched with syntactic features and an unsupervised word alignment method in the alignment approach. During alignment, first the IBM word alignment model [99] linearizes AMR graphs, then the nodes of the graphs are mapped to English sentences. [100] presents an AMR aligner for Portuguese, which is a morphologically rich language. The authors use the Word Mover's Distance [101] and lexical lists for the alignment of abstract concepts and entity names.

5.1.2.2 Semantic role labeling

Early SRL approaches were based on linear classifiers [102,103], which are highly dependent on manually generated features developed based on syntax. With the

proliferation of neural network models, syntactic features were adapted to these models [104]–[109]. Later studies have focused on building SRL systems with no or a small syntactic directive. [110]–[112]. Recently, end-to-end solutions for SRL have attracted extensive interest, and this kind of model has surpassed the previous state of the art. [113] has developed an end-to-end model with an affine attention mechanism, [114] has proposed an end-to-end model for both dependency and span SRL with a unified argument representation, [115] has adapted BERT [116] to the SRL task according to [114]. On the other hand, there are several studies on SRL in Turkish. [51], [52] have made efforts to build the first Turkish PropBank and [117] has introduced a char-LSTM based SRL tool.

5.2 Rule-based Parser

We develop a rule-based parser inspired by [118] the authors proposed an AMR parser built on top of dependency parsing. The proposed parser takes the output of the dependency parser of a sentence and converts it into an AMR graph. The parser is designed using the supervised approach, which requires the alignments of concepts to words in sentences. It is at this point that our parsing approach deviates from the [118] due to the following limitations and difficulties. (1) When we developed the parser, we only had 100 AMR-annotated sentences available during the development of the parser (at the first iteration of our MAMA cycle described in Chapter 4), the size of the annotated data is not as large as it needs to be for training. (2) No aligner was available[1] and developing an aligner from scratch was not trivial due to the complex Turkish morphology. Given our limitations, we utilize an unsupervised approach that maximizes the use of available resources (e.g., the Turkish PropBank), as well as hand-crafted lists that are most appropriate for our problem.

Our parser is designed such that its actions are determined by a rule set. The rule set contains the parsing rules that select the next action during parsing, and the mappings of sentence components to AMR concepts. A sentence component can be a morpheme that should be represented by a concept such as *-sIz*, *-li*, etc., a word that invokes abstract concepts such as ‘yıl’ (*year*) that invokes ‘temporal-quantity’,

[1]The proposed aligner had not yet been developed at this stage of the study.

'Istanbul' that invokes 'city' and 'name', etc. The mapping includes the semantic features of sentences, which enables the parser to cover the compositional semantics defined at the morphological level. The parser uses these features by combining them with the syntactic features and representing verbs at both word and morphological levels. In agreement with the literature, we call this mapping alignment. In what follows, we use the terms 'mapping' and 'alignment' interchangeably. In Turkish, words may correspond to complex AMR graphs, and the integration of such complex structures requires additional action during the parsing phase. Also, several suffixes have multiple meanings, and their uses and functions must be distinguished when selecting the correct alignments of such components from the rule list. Syntactic features such as dependency relations and morphological tags are useful resources for this task. Therefore, the parser performs AMR graph construction and selection of correct alignments between tokens and AMR concepts simultaneously. Figure 5.1 provides the different alignments of the same word 'yıllardır' depending on its usage. The alignment on the left shows the alignment of this word when it is used in the sentence as 'for years', the right side shows the alignment when it means 'these are years'.

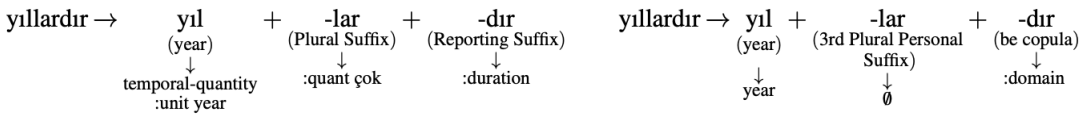

Figure 5.1 : An alignment of the word 'yıllardır' according to its usage.

The parser is highly dependent on morphological features. A suffix may correspond to a concept or a relation, requiring morphological analysis. Therefore, use the Turkish PropBank [52] as input for developing the parser. The Turkish PropBank builds on the IMST by adding a semantic layer. The parser takes its input in CoNLL form [2] and generates its AMR graph. An example of the parser's input can be found in Table 5.1. The first 7 columns are from IMST and the last column was added at PropBank annotations.

[2]Universal Proposition Banks https://github.com/System-T/UniversalPropositions.

Table 5.1 : A sentence “Bu ilişkiyi bitirelim, böyle yürütemeyeceğim, dedi.” (*Let’s end this relationship, I can’t run it like this, she said*) in the Turkish PropBank.

Ind.	Word	Lemma	PoS	Morphological Features	Head	Dependency Rel.	Semantic Layer				
1	Bu	bu	Det		2	DETERMINER					
2	ilişkiyi	ilişki	Noun	_	3	OBJECT	-	-	‘A1’	‘A0’	-
3	bitirelim	bitir	Verb	A3sg-Pnon-Acc	6	COORDINATION	‘Y’	‘bit.01’	-	-	-
4	,	,	Punc	Pos-Opt-A1pl	3	PUNCTUATION					
5	böyle	böyle	Adv	_	6	MODIFIER	-	-	-	‘AM-MNR’	-
6	yürütemeyeceğim	yürü	Verb	Caus-Able-Neg-Fut-A1sg	8	OBJECT	‘Y’	’yürü.01’	-	-	‘A1’
7	,	,	Punc	_	6	PUNCTUATION					
8	dedi	de	Verb	_	0	PREDICATE	‘Y’	‘de.01’	-	-	-

The features provided by the dataset with their column numbers are as follows:

- word position within the sentence in the 1st column.
- word surface in the 2nd column.
- word lemma in the 3rd column.
- PoS tag in the 4th column.
- morphological feature in the 5th column.
- dependency head index in the 6th column.
- dependency relation in the 7th column.
- semantic role labeling features in 8th column.

One should note that we show the condensed form of the example in Table 5.1. In the original format, there are repetitive columns, and the last column shows the multiple columns that indicate the arguments of a given predicate in the order in which they appear in the sentence. The annotation ‘Y’ indicates that the following tag is a verb frame. The Turkish PropBank is of great importance for the parser, since it provides the frames with their arguments. With its help, we try to handle the verbs that require investigation at the morphological level (i.e., x-rooted HPVs) by adding syntax-aware rules to our rule set. For example, xlAn frames are represented by either ‘ol.04’ (to get) when x is a noun, or ‘ol.02’ (to become) when x is an adjective.

The parser consists of two steps: constructing a tree, which we call an 'interstep tree', and converting the interstep tree into an AMR graph. We explain these steps in the following subsections, and the parser is available in GitHub project[3] for further studies.

5.2.1 Inter-step tree

The inter-step tree is built by merging the nodes and relations of the dependency tree with the frames and relations of the PropBank. The parser takes an input example I in CoNNL format, where I is defined as I=(V, A, *morph*, t, Prop). The definition of the variables is as follows:

- V = $\{v_i \mid \text{i} \in [0,\text{n}], \text{i} \in \mathbb{N}\}$ is a set of nodes representing word tokens in the sentence[4],
- A = $\{a_{ij} \mid \text{i,j} \in [0,\text{n}], \text{i} \neq \text{j}, \text{i,j} \in \mathbb{N}\}$ is a set of dependency relations between nodes v_j (the head) and v_i (the dependent),
- *morph* contains the morphological features of the words,
- t is parts-of-speech tags of the words,
- Prop = $\{prop_{ik} \mid \text{i} \in (0,\text{n}], \text{i} \in \mathbb{N}^+, \text{k} \in [0,\text{m}], \text{k} \in \mathbb{N}\}$ represents a set of tags of the semantic layer, where $prop_{ik}$ corresponds to the k^{th} annotation of node v_i and m is the number of semantic layer tags that node v_i has.

We define the inter-step tree D = (C, R, *NodeProperties*), where C = $\{c_i \mid \text{i} \in (0,\text{n}], \text{i} \in \mathbb{N}^+\}$ represents a set of nodes, R = $\{r_{ji} \mid \text{i,j} \in (0,\text{n}], \text{i} \neq \text{j}, \text{i,j} \in \mathbb{N}^+\}$ represents a set of edges and *NodeProperties* contains a quadruple consisting of the properties (*morph*, t, head node, dependency relation) of each node c_i. c_i and r_{ij} are defined as follows (Equation 5.1 and Equation 5.2), where orderof(j) is a function indicating the order of the predicate within the sentence. The smallest index that the argument roles can take is 2, since k=0 and k=1 are reserved for predicate declaration shown in Table 5.1.

[3] `https://github.com/amr-turkish/turkish-amr-parser`.

[4] v_0 is the root node in the dependency tree, it is not present in the sentence.

$$c_i = \begin{Bmatrix} prop_{i1} & \text{if } prop_{i0} = \text{Y} \\ v_{\text{i}} & \text{otherwise} \end{Bmatrix} \tag{5.1}$$

$prop_{ik}$ can have three values, which are a verb frame, an argument role, and the letter 'Y', and can be expressed by a node or relation in the inter- step tree, depending on its value. In cases where $prop_{ik}$ has multiple argument roles, the very first role is used as c_i, and the others establish reentrancy connections (details in the section 5.2.2). The dependency components are used directly in the construction of D if they do not have semantic level tags.

$$r_{ij} = \begin{Bmatrix} prop_{ik} & \text{if } k = \text{orderof}(j)+1 \\ a_{ij} & \text{otherwise} \end{Bmatrix} \tag{5.2}$$

Figure 5.2 shows the inter-step tree of the sentence "Bu ilşkiyi bitirelim, böyle yürütemeyeceğim." (*Let's end this relationship, I can run it like this, she said*) given in Table 5.1. The verb frames 'bit.01' (*end*), 'yürü.01' (*walk*), and 'de.01' (*say*) are the semantic layer tags for the word lemmas 'yürü', 'bitir', and 'de', respectively, and are used in the construction of the inter- step tree.

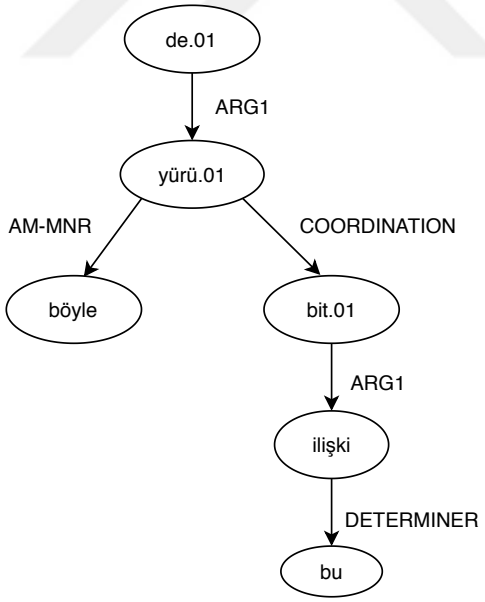

Figure 5.2 : Inter-step tree and AMR graph for "Bu ilişkiyi bitirelim, böyle yürütemeyeceğim, dedi." (*Let's end this relationship, I can't run it like this, she said*).

The semantic relation tags (AMR-MNR, ARG1) are replaced by their dependency equivalents. The other nodes ('bu' (*this*), 'ilişki' (*relationship*)), and relations (DETERMINER, COORDINATION) are directly copied into the tree. As you can see, the word 'ilişki' (*relationship*) has two argument relations A0 (ARG0) and A1 (ARG1) (see Table 5.1), and A1 is used in the inter- step tree since it is the first tag.

5.2.2 Tree-to-graph conversion

A similar notation to [118] is adopted to define our parser; however, it diverges from the mentioned study with its actions and alignment strategy.We define our rule-based tree-to-graph parser as

$Cr = (Cr, Actions, Cr_0, Rules)$.

- Cr is a set of parsing states which contains <D,q> couples,
- $Actions$ is a set of actions A: Cr→Cr,
- Cr_0 is an initialization step where inter-step tree is built,
- $Rules$ is a set of conversion rules.

A parsing state has D and q, where q is a queue containing word indexes corresponding to the sentence order, D is inter- step tree defined above. The graph conversion starts with the construction of the inter-step tree, then the parser processes all the nodes of D using the word indices contained in q. It takes the first element of q and iterates, specifying the corresponding node in D. The node properties provided by D are processed and the next action is determined according to the *Rules*. At each iteration, the set *Rules* returns a set of actions according to the given node properties ($[Rule(c_i, NodeProperties_i) \rightarrow Actions_a]$), and the parser applies the action to D.

We have eight types of actions shown in Table 5.2 that covering all possible situations in the conversion process. $Pr(i)$ returns the parent index of a node at index i, $Ch(i)$ returns all the children indexes of a node at index i. $\gamma : C \rightarrow R$ and $\zeta : C \rightarrow R$ are the functions responsible for relation operations. The γ function establishes an AMR relation between two input concepts, where the second argument of the function

becomes the parent node after the action, ζ deletes the relations between the current node and its parent. The function takes the current node and its parent in focus as arguments, since the initial inter-step tree is constructed from the dependency tree. The dependent could have only one head initially, but multiple heads when the AMR graph is constructed.

Table 5.2 : Actions.

Action Name	**Action Performed**	**Assigned labels**
Add Edge	$\gamma[(c_{q_i}, c_j) \rightarrow \{r_{c_{q_i}c_j}\} \cup R], c_j \in C$	$\iota[(r_{c_{q_i}c_j}) \rightarrow l], l \cup L$
Delete Edge	$\zeta[(c_{q_i}, c_{Pr(q_i)}) \rightarrow R \setminus r_{c_{q_i}c_{Pr(q_i)}}]$	None
Add Node	$\delta[(c_{q_i}) \rightarrow \{c_k\} \cup C, \gamma(c_k, c_{q_i})]$	None
Delete Node	$\varphi[(c_i) \rightarrow C \setminus c_i]$	None
Replace Head	$s = Ch(Pr(q_i)), c_k \cup C, \gamma(c_j, c_k), j \in s, \gamma(c_k, c_{Pr(Pr(q_i))}), \varphi(c_{Pr(q_i)})$	$\iota[(r_{c_jc_k}) \rightarrow l], l \cup L$
ReAttach	$\zeta(c_{q_i}, c_{Pr(q_i)}), \gamma(c_{q_i}, c_k), c_k \in C$	$\iota[(r_{c_{q_i}c_k}) \rightarrow l], l \cup L$
Swap	$\zeta(c_{q_i}, c_{Pr(q_i)}), \gamma(c_{Pr(q_i)}, c_{q_i})$	$\iota[(r_{c_{Pr(q_i)}c_{q_i}}) \rightarrow l], l \cup L$
Merge	$c_k = c_{q_i} \cup c_{Pr(q_i)}, \gamma(c_k, c_{Pr(Pr(q_i))}), \varphi(c_{q_i}), \varphi(c_{Pr(q_i)})$	$\iota[(r_{c_kc_{Pr(Pr(q_i))}}) \rightarrow l], l \cup L$

$\delta : C \rightarrow C$ creates a new concept node from an existing node based on its morphological features. It also establishes a relation between the new node and the current node. $\varphi : C \rightarrow C$ deletes a node given as argument. $\iota : R \rightarrow L$, where L is the AMR relation set, assigns a label to the given edge as argument. The eight actions are explained below along with their illustrations. The red color indicates the part of the inter- step tree that is modified by each action, and the black dashed arrows indicate other components not shown in the figure.

- **Add Edge**: It simply adds an edge between the node in the queue with index i (c_{q_i}) and the other node with index j (c_j) in the inter- step tree shown in Figure 5.3. The newly created edge $r_{c_{q_i}c_j}$ is included in the edge set R. Also, $r_{c_{q_i}c_j}$ is assigned a label l from the AMR label set L.

- **Delete Edge**: It deletes the edge between the node in the queue with index i (c_{q_i}) and its parent, as shown in Figure 5.4. The removed edge $r_{c_{q_i}c_{Pr(q_i)}}$ is excluded from R.

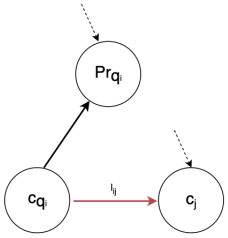

Figure 5.3 : Creating the edge l_k between c_{q_i} and c_j.

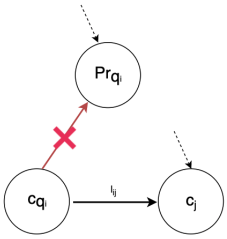

Figure 5.4 : delete the edge between c_{q_i} and its parents. The red x is the character for the deletion

.

- **Add Node**: It creates a new node c_k based on the node in the queue with index i (c_{q_i}) and establishes an edge between c_k and c_{q_i} where the parent is c_{q_i}. Figure 5.5 shows the action. The newly created node c_k is added to the node set C.

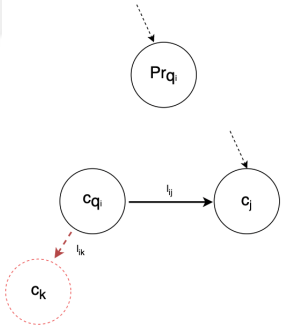

Figure 5.5 : Adding new node c_k to the inter- step tree as a child of c_{q_i}. The dashed line indicates he newly created node and its connection.

- **Replace Head**: As indicated in the Figure 5.6, the node in the c_{q_i} queue is replaced by a new onec_k (it is c_m in Figure 5.6). It first takes all children nodes of the node c_{q_i}, then creates edges between the children and c_k. The newly created node c_k is included in the node set C and c_{q_i} is excluded from C.

- **ReAttach**: It deletes the edge between a node in the queue (c_{q_i}) and its parent. A new edge is established between c_{q_i} and a node c_k. Figure 5.7 shows the action.

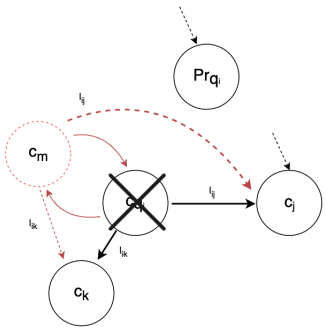

Figure 5.6 : The newly created node c_m (the one with the red dots) is replaced by c_{q_i} and inherits its children (c_j and c_k). The black cross indicates that c_{q_i} is removed.

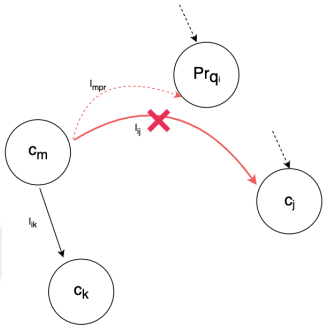

Figure 5.7 : Deleting the edge between c_m and one of its parents c_j. The new edge is created between c_m and $c_{Pr(q_i)}$.

- **Swap**: It deletes the edge between a node in the queue (c_{q_i}) and its parent. It creates a new edge between these two nodes in the opposite direction. Figure 5.8 shows the action.

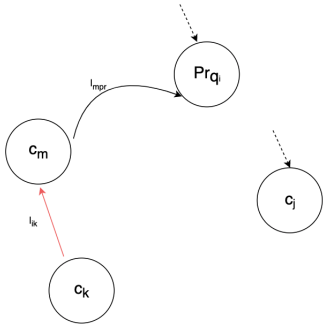

Figure 5.8 : A new edge between c_m and its parent c_k in the opposite direction.

- **Merge**: a new node c_k is created by combining the node in the queue with index i (c_{q_i}) and its parent, and c_k is joined to the grandparent of c_{q_i}. The nodes c_{q_i} and $c_{Pr(q_i)}$ are removed from the node set C. Figure 5.9 shows the action.

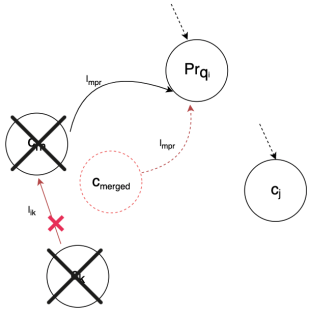

Figure 5.9 : Merging of c_k and its parent c_m. The newly formed node c_{merged} is connected to $c_{Pr(q_i)}$.

The parser processes q twice consecutively. These processes: (i) add the missing nodes (i.e., the nodes coming from the morphology) to D, (ii) delete the redundant nodes (i.e., the nodes corresponding to words that do not contribute to the sentence meaning), (iii) add reentrency relations, and (iiii) convert the trees between each step into AMR graphs. We call these two processes graph conversion and post-processing, respectively.

Graph conversion consists of three consecutive sub-steps which are explained with the demonstrations by using the sample provided in Table 5.1. The following items give the explanations:

- Node removal is the substep of removing words that do not contribute to sentence meaning. Such words usually take the role of determiners or intensifiers in the sentences. Therefore, the parser simply removes the nodes associated with the relations.DETERMINER and INTENSIFIER in the inter- step tree indicated in Figure 5.10. However, it is worth to mention that it is not the proper approach to generalize this, since the meaning contribution of words with such roles depends on their usage and the overall sentence meaning. Unfortunately, the rule base parser is not able to distinguish which of them should be removed and which should not.

- The addition of reentrancy relations (i.e., reentrancies (explained in the Subsection 2.1.2)) to the inter-step tree turns the tree into a graph. The nodes with multiple relations, which we call 'reentrancy nodes', usually have more than one semantic role (Table 5.1 node at the 2^{nd} index). As we mentioned in the previous subsection,

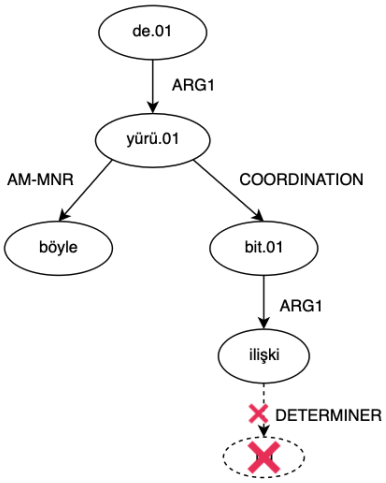

Figure 5.10 : The parser removes the node “bu” since its dependency connection is “DETERMINER”. The red x and dashed lines indicate the removed components.

the first tag is embedded in the inter- step tree. The others join the parsing in this step. The parser establishes new relations between reentrancy nodes and the most appropriate nodes selected by the rule set. In Figure 5.11, it is shown that the previously absent relation *ARG0* (A0) is added in D between ‘ilişki’ (*relationship*) and ‘yürü.01’ (*work*). As a result of this step, the inter- step tree turns into a graph.

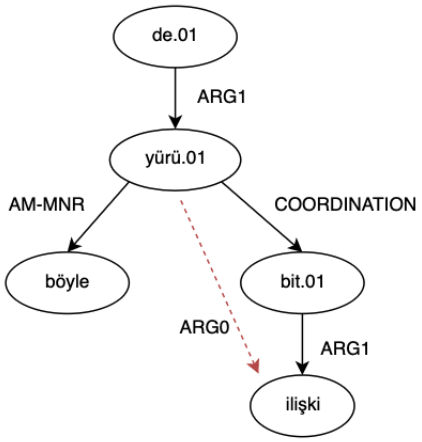

Figure 5.11 : The parser adds the reentrancy relation ARG0 between ‘ilişki’(*relationship*) and ‘yürü.01’ (*walk*).

- The concepts arising from morphology are the most essential parts of Turkish AMR parsing. As we discussed in Chapter 3, the majority of the meaning contributions come from these suffixes. The parser uses the given morphological properties of

nodes (*Node-Properties*) and it may perform node or relation additions. Table 5.3 briefly describes the operation that the parser can perform in this step.

Table 5.3 : Parser operations of handling Morphology based nodes

Op No#	Op	What is done?
1	Adding a null subject	Creating a concept for personal suffix, connecting it to the node in focus with *ARG0*.
2	Adding modalities	Creating a concept for the modality marker, connecting it to the node in focus with *ARG1*.
3	Adding negativity	Connecting node in focus with constant minus with relation *:polarity*.
4	Handling Voice	Adding 'yap.03' and connecting to the node in focus with *ARG1*.
5	Handling case markers	Connecting the node in focus to its dependency head with *:location*.

At the end of the graph conversion step, the previously absent nodes 'biz' (*we*) and 'o' (*s/he*) are revealed by personal suffix markers extracted from *NodeProperties* and become the agents of the predicates 'bitire**lim**' and 'dedi'. The word 'yürü**temey**eceğ**im**' (the verb *run*[5] in the future tense with modality and negativity markers) has one causative suffix and multiple inflectional suffixes (i.e., modality, negativity and personal markers). The parser adds the concepts 'yap.03' (*make*), 'mümkün.01' (*possible*), '-' (*minus*) and 'ben' (*I*) to represent causativity, modality, negativity and the agent who does the action respectively. It should be noted that the word 'bitir' (the verb *end*) is constructed from the root word 'bit' (*to end*) by the causative suffix *-ir*. However, since the morphological analyzer outputs its lemma as 'bitir' instead of 'bit' and misses to output the causative structure (Table 5.1 node at the 3^{rd} indice), our AMR parser fails to extract this information from the node properties and to add the 'yap.03' (*make*) concept in this example to represent causativity. Figure 5.12 gives the output of the graph conversion step.

[5]The verb 'yürüt' is constructed from the verb 'yürü' (*walk*) by the causative suffix *-t* and gains the meaning of 'make it work'. However, its meaning in this sentence can be translated as '*run*' ('to run a relationship').

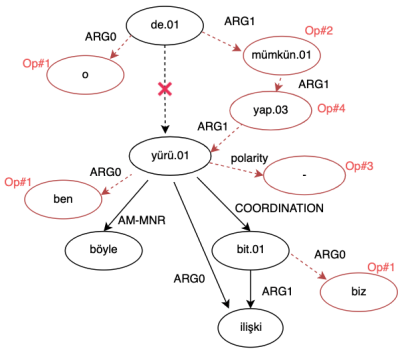

Figure 5.12 : Inter- step tree after graph conversion step. The red dashed components indicate the additional concepts and relation and Op# indicates the operation that created them.

In post-processing, non-AMR components that could not be mapped in the previous phase are mapped to AMR concepts and relations. Some words invoke abstract concepts that should be added to D. The nodes representing such concepts are aligned with the abstract concepts and their AMR representations are integrated into D. On the other hand, the mapping of relations can be either a renaming of edges or a transformation of an edge into a corresponding AMR subgraph. If the AMR specification has a relation that has the same meaning, renaming the edge is quite simple, as shown in Table 5.4. Otherwise, the parser triggers an action sequence that converts the relation to an equivalent AMR subgraph.

Table 5.4 : Direct mapping of ProbBank relations to AMR relations.

PropBank	AMR	PropBank	AMR
AM-MNR	:manner	AM-LOC	:location
AM-ADV	:mod	AM-TMP	:time
AM-INS	:instrument	AM-COM	:accompanier

In Figure 5.13 you can see the final graph produced by our parser. It maps AMR-MNR to the relation :manner and turns COORDINATION into a subgraph to which the node ‘and’ is added. Note that our parser is developed mainly based on the syntactic features of words and is not able to capture the deep semantic meaning of the sentence. In Figure 5.13, we see that the parser fails to construct the relation *:cause* since it could not get any hint about this semantic relation from the node features.

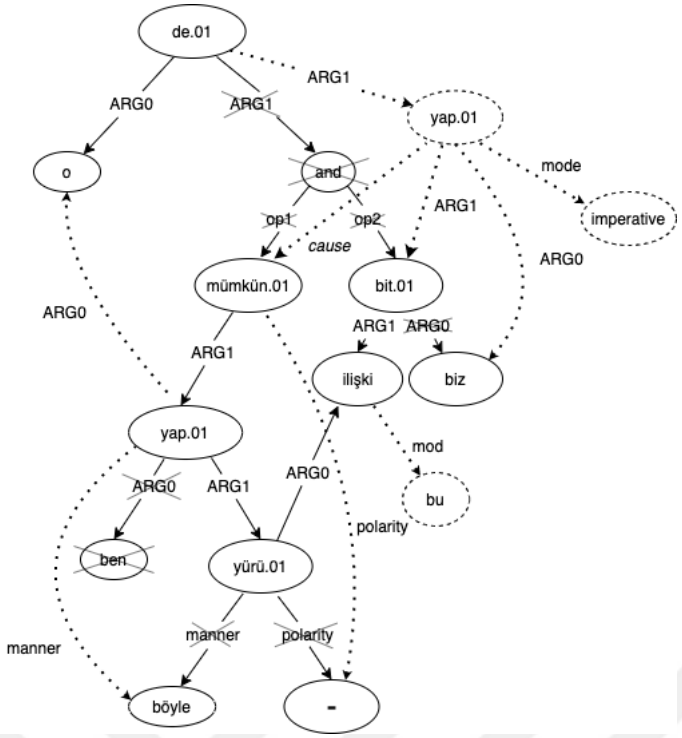

Figure 5.13 : The final AMR graph of the sample sentence given in Table 5.1, The concepts and relations generated incorrectly by the parser are depicted with a cross sign on the concept or relation, and their correct counterparts are indicated with dashed lines.

5.2.3 Evaluation

We evaluate the effectiveness of the parser by 1) comparing its outputs to gold standards and 2) using it for semi-automatic annotation.

The rule-based parser achieved a Smatch Score of 0.65 and 0.60 (on the Turkish AMR corpus in Chapter 4) at the end of the first and second MAMA cycle iterations, respectively. One should note that the rule-based parser is designed for use with gold-standard dependency and PropBank annotations. In practice, its performance is affected by the errors introduced by automatic morphological analysis, dependency parsing, and semantic role labeling. Nevertheless, we believe that this first Turkish AMR parser will act as a strong baseline for future studies on Turkish AMR parsing. As we detailed in Chapter4, our corpus contains 600 sentences having gold-standard dependency and PropBank annotations (IMST), where the parser's performance is measured to be 0.6. The other 100 sentences does not have automatically generated

dependency and PropBank annotations (tLP) and the parser's performance is measured to be 0.54. We also measure its performance on the short and complex sentences, whose distribution in the corpus can be found in the corresponding section. The parser's Smatch Score is 0.75 on short sentences and 0.58 on complex sentences.

For the second set of evaluations, we create two experimental setups to measure the effects of the parser on the annotation process. First, we select two sets of 10 sentences from IMST with similar syntactic and semantic structures. The selected sentences are also similar in terms of sentence length and structural complexity. We then record the time spent by a single human annotator annotating these two sentences separately. For one of the sets, the annotation is done from scratch; for the other, it is done by semi-automatic annotation, where the experienced human annotator corrects the output of the rule based parser. The time required for both annotation methods is given in Table 5.5. From this it can be seen that the annotation time is remarkably reduced (of around two-thirds) when the parser is used as a preprocessor and the human annotator corrects its outputs instead of annotating from scratch (manual annotation). It should be noted that the selected sentences were not very difficult and the time required to annotate a single sentence cannot be generalized.

Table 5.5 : Annotation times.

Annotation Style	**Total Time (in sec)**
Manual Annotation	1438
Semi-automatic Annotation	**545**

In the second experiment, we randomly select 25 additional sentences from IMST that were not previously annotated. Two human annotators annotate these sentences, one from scratch and the other on the outputs of the rule-based parser. The inter-annotator agreement between the two human annotators is measured as 0.85 Smatch scores. We also perform an error analysis on the sentences where there is no agreement between our annotators, and find that the annotations of the human annotator working on the parser's outputs match the predicate frame names better than the annotations of the human annotator working from scratch. This result is to be expected since our parser uses gold-standard predicate frame tags that should be replaced by an automatic predicate disambiguator in a real scenario, while the human annotator working from

scratch tries to select them manually each time, which is error-prone. On the other hand, we see that the parser instructs the annotator to use more conjunctions (as illustrated in the previous section Figure 5.13 the use of 'and' instead of the relation *:cause*) and possessiveness (instead of *:topic*, *:part-of*, etc.) in the complex sentences than necessary. The parser also helps extensively the human annotator in cases that morphologically originated (e.g., null subject, dropped pronouns, modality) and that the human annotator might miss when working from scratch.

5.3 Data-Driven Parser

In contrast to the literature, where studies mainly focus on neural approaches, we use the techniques of ML for our data-driven parsing approach. We choose ML techniques because they consume a relatively small dataset compared to neural methods. The Turkish AMR corpus contains 700 sentences, which is not sufficient for training a neural model. Our goal is to investigate the answers to the following questions: (i) Can a data-driven parser be developed using the Turkish AMR corpus, which is insufficient to train a data-driven parser? and (ii) What are the challenges of Turkish AMR parsing?

Our data-driven parser (DDP) takes an input sentence $I = w_1, w_2, ..., w_n$, where w_i is a token in the vocabulary, and outputs an AMR graph $G = (C, R)$, where C represents concepts within the graph and R describes the relations between concepts. Each concept and relation has a label of L_c and L_r, respectively. The system consists of two sequential steps, namely the identification of concepts and relations, following the general approach in this domain. For each step, we train independent classifiers, each of which has its own functionality and relies on a different set of features.

The feature sets are developed considering Turkish morphology and AMR specifications. We create features based on the semantic role labels, syntactic dependencies and their combinations with PoS features. All features used can be found in Table A.1 and Table A.2 in Appendix A. To reduce sparsity, we also use pre-trained word embeddings whose significance has been proven in many NLP tasks. We use

fasttext[6] [119] vectors. It is worth mentioning that our features (lexical, positional and morpho-semantic) are the outputs of the other NLP subtasks.

5.3.1 Concept identification

The concept identification step is formulated as a sequence labeling task and illustrated in Figure 5.14 that shows the same sentence used for examplifying RBP. The parser

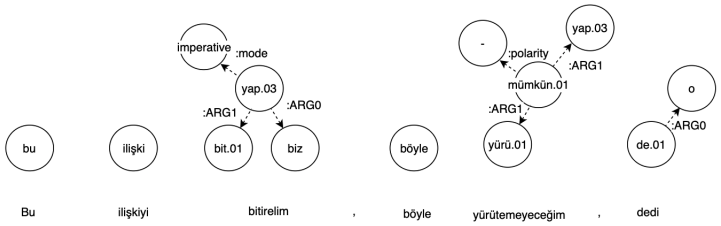

Figure 5.14 : The illustration of the concept identification step the DDP. This step aims to find only concepts, relations will be treated in the next step, therefore they are represented with dashed lines.

identifies the concepts corresponding to spans of words (or a single word) within the sentence[7]; a word span word may label with multiple concepts or an empty set. We reformulate this step as nine separate classification tasks for finding different concept types, described in Table 5.6[8]. Additionally, as we described in Chapter 3, some suffixes have specific meanings, and can be directly assigned to the predefined concepts (e.g., the modality marker *-Abil* and its corresponding concept 'mümkün.01', the negativity marker *-mA* and its constant concept '-', etc.). This map allows us to identify concepts without a learning process. We add a post-processing layer, where morphology-drived concepts created. It should be noted that there may be cases where a word may be associated with conflicting concepts. We do not intend to prevent this situation and leave the resolution of this situation to the next parsing step.

5.3.2 Relation identification

The relation identification constructs edges between concepts identified in the previous step. We conceptualise this task as extracting a subgraph from the fully connected

[6]https://fasttext.cc/docs/en/crawl-vectors.html

[7]The spans of words have to be aligned with the concept fragments to train the parser, the aligner for Turkish is described in Section 5.4.1.

[8]The features used in the training of these classifiers are available in the Table A.3 in the Appendix A.

Table 5.6 : The classifiers used in concept identification. Their functionalities are given in the description column.

Classifiers Name	Description
Type classifier (*TC*)	This is a multiclass classifier that predicts the created concept will be a verb frame, a lemma, or an empty concept. If it is a verb frame, it tries to predict its meaning.
Person classifier (*PC*)	This is a multiclass classifier that predicts pronoun concepts to be created for pro-drop pronouns and null subjects.
Person/Thing classifier (*PTC*)	This is a multiclass classifier that predicts additional person/thing to be created for representing headless relative construction.
Abstract Classifier (*AbsC*)	This is a multiclass classifier that predicts the abstract concept to be created. This classifier does not include named entities.
Invoke Classifier (*IC*)	This is a multiclass classifier that predicts the additional verb frames invoked by the words. For example 'benze.01' may be invoked by the word 'gibi' or *-sIz* may invoke additional verb frame 'yok.01'.
Conjunction Classifier (*ConjC*)	This is a multi-class classifier that predicts if a word invokes the concept 'and'.
Cause Classifier (*CauC*)	This is a binary classifier that decides whether the concept 'ol.17' should be created.
Question Classifier (*QC*)	This is a binary classifier that decides whether the concept 'amr-unknown' should be created or not.
Named entity Classifier (*NeC*)	This s a multi-class classifier that predicts entity types so that the related concepts can be created.

simple graph. We assume that all concepts identified in the previous step are interconnected. The aim is to find a subgraph whose edges have edge labels. This approach is a modification of the Minimum Spanning Graph algorithm [67], where the edges have numerical costs. In our approach, we replace the numerical values with edge labels controlled by L_r. If an edge has no label, its value is infinity, otherwise it is 1. Our edge labeler consists of three classifiers (argument relations classifier (*ArgC*), non-core relations classifier (*nonCC*), constant relations classifier (*constC*)). The argument relations classifier is responsible for finding *:ARGx* relations. Despite

the SRL relations, we need this classifier to identify the arguments of the invoked predicates by nominals or morphemes. Such predicates do not exist in the sentences, so their arguments are not included in SRL. In addition, AMR has the concept of inverse relations, which is not the case with SRL. The other classifiers identify non-core and constant relations of AMR.

The parser attempts to assign a label to an edge by providing concept pairs to the classifiers. A classifier takes feature vectors[9] of a concept pair $<c_i, c_j>$, outputs $l_{rij_{clf}}$ which is a relation label from l_r and a confidence score. $l_{rij_{clf}}$ may be $\emptyset$, meaning that there is no relation label between c_i and c_j. Ideally, an edge may have only one label prediction, and the other classifiers predict as $\emptyset$, but edges with more than one relation may appear. In these cases, we choose the one with the highest confidence value. It should be pointed out that the confidence values of three separate classifiers are comparable when they are calibrated.

5.3.3 Experiments and evaluation

We train and evaluate the DDP on IMST and tLP parts of the Turkish AMR corpus. For each parsing step, we report the performance of the parser by giving the classification results in terms of precision, recall and F1-score. We evaluate the performance of the full parser using the Smatch score as in RBP. For training, we clean up the training set by removing sentences longer than 30 words, since the aligner tends to make more frequent errors when aligning long sentences, leading to incorrect predictions. Although the problematic examples are removed from the dataset, the training still suffers from (1) insufficient training set size (i.e., 564 sentences) and (2) class imbalance. [120] shows that textual data augmentation significantly improves performance regardless of task, models, and dataset size. However, textual augmentation has not yet been applied to AMR parsing and does not guarantee the elimination of the class imbalance problem. Considering the results of the aforementioned study, we follow a simple oversampling strategy that only aims to avoid underfitting. We use the Synthetic Minority Oversampling Technique (SMOTE) [121], which generates new samples on lines drawn between closely spaced samples

[9] Table A.4 in Appendix A indicates the features used in the training of these classifiers.

in the feature space. Since tLP as a test set does not have a gold standard annotation, we use assistance tools (e.g., Semantic Role Labeler, Named Entity Recognizer, Dependency Parser, etc.) to extract its features. The tools used in this phase are described in section 5.4. In our experiments, we use four different classifiers: lightGBM [122], logistic regression [123], XGBoost [124], and random forest [125]. In the first step, we trained all four classifiers separately for each classification task and investigated their performances. The results are given in Table 5.7 as average macro F1 scores[10].

Table 5.7 : The average macro F1 scores achieved by different classifiers in the Concept Identification Step.

Classifier	*TC*	*PC*	*AbsC*	*IC*	*ConjC*	*CauC*	*QC*	*NeC*
Random Forrest	**0.23**	0.39	0.27	0.60	**0.77**	0.50	0.93	**0.50**
LightGBM	0.06	0.52	0.34	0.55	0.70	**0.60**	0.87	0.49
XGBoost	0.20	**0.58**	0.25	**0.61**	0.75	0.59	0.87	0.49
Logistic Regression	**0.23**	0.23	**0.39**	0.56	0.67	0.57	**0.94**	0.49

The relation identification step is evaluated by measuring the overall performance of the three classifiers described in the previous subsection (Section 5.3.2). As [52] has been reported, logistic regression has significant performance in learning both discrete and continuous features in determining semantic relations between words (i.e., SRL), therefore we train logistic regression for *ArgC* differently from the other two where LightGBM is used. The relation identification system achieves 0.39 micro-F1 scores. Table 5.8 gives the precision, recall, and F1 scores for each relation achieved by the parser. To measure the overall performance of our parser, we selected the best performing classifiers for the entire parsing pipeline [11].

The parser achieves a Smatch score of 0.17 F1, which is far behind the result of RBP already has achieved. We identify the possible reasons for this extremely low score and list the major ones:

1. **The errors propagated through the concept identification step**: We group these errors into three subcategories: (1) The abstraction layer of AMR requires

[10] Since there is no sample for *PTC* in the test set, we could not evaluate this classifier
[11] We train logistic regression for *PTC*.

Table 5.8 : Precision, Recall and F1 scores achieved for each relation in the Relation Identification Step

Feature	P	R	F1	# of Sample
:ARG0	0.86	0.62	0.72	167
:ARG0-of	0.72	0.58	0.64	176
:ARG1	0.71	0.44	0.54	152
:ARG1-of	0.52	0.48	0.50	178
:ARG2	0.88	0.17	0.29	40
:ARG2-of	0.53	0.23	0.32	43
:ARG3	0.83	0.71	0.77	7
:ARG3-of	0.67	0.57	0.62	7
:ARG4	0.33	0.67	0.44	3
:ARG4-of	0.67	0.67	0.67	3
:degree	0.57	0.19	0.29	21
:direction	0.00	0.00	0.00	4
:domain	0.00	0.00	0.00	13
:duration	0.00	0.00	0.00	6
:location	0.60	0.30	0.40	20
:manner	0.57	0.18	0.28	22
:mod	0.31	0.21	0.25	68
:mode	1.00	0.83	0.91	12
:name	1.00	0.43	0.60	7
:opN	0.55	0.63	0.59	82
:ord	0.00	0.00	0.00	5
:part-of	0.00	0.00	0.00	4
:polarity	0.93	0.59	0.72	22
:poss	0.15	0.21	0.18	38
:quant	0.43	0.14	0.21	22
:time	0.61	0.48	0.54	23
:unit	1.00	0.56	0.71	9
:value	0.00	0.00	0.00	5
O	0.53	0.80	0.64	615

deriving abstract concepts from some words (e.g., 'sabah' (*morning*) derives *temporal-quantity*), in which case the parser systematically fails. There are more than 10 abstract concepts in AMR, which corresponds to a large number of words that need to be abstracted. Our methods for identifying abstract concepts do not cover all cases. (2) The AMR has several entities, unfortunately the parser can only recognize 3 of them (i.e., location, organization, person). (3) The little prince does not have gold standard SRL labels. Therefore, the errors of the semantic role labeler directly affect the parsing, since our classifier relies heavily on the SRL features.

2. **The errors propagated through the relation identification step**: The main reason for the low Smatch score of the parser is the errors that occurred in the relation identification step. We have observed that the parser fails to learn almost all relations. We believe that the lack of golden SRL labels, the small size of the training data, and the inability of this parsing method to capture semantic relations between words are the main reasons for this under-fitting.

3. **The errors propagated through the alignment step**: The outputs of the aligner are directly used for training the parser, therefore, the systematic errors (e.g., the misalignment of the light verbs) made by the aligner cause the parser to learn from erroneous data. We note that the misalignment of conjunctions makes it difficult for the parser to learn *:opN* relations

To sum up, we observed that building a data-driven parser requires a large size of training set where all semantic properties of Turkish are included, which shows that increasing the number of annotated data is a must. Furthermore, having strong assistance tools performing close to gold standard annotations is another important point of data-driven parsing. Increasing their performance certainly affects the overall parsing, but we believe it is more useful to take approaches that eliminate such tools in parsing rather than striving to improve them.

5.4 Assistant Tools

The fact that the AMR contains semantic aspects of the sentence may emerge the need of usage additional resources in the parsing. These resources may be either corpora annotated at different levels (e.g., PropBank [7] and AMR annotated corpora (e.g., LDC AMR corpora)) or other NLP tools such as tokenizers, parts-of-speech taggers, syntactic analyzers, named-entity recognizers, linkers, or semantic role labelers. The parsers explained in this study rely heavily on the semantic and syntactic features extracted using such tools (e.g., dependency parser, semantic role labelers, lemmatizers, etc.). ITU NLP pipeline [126] provides some of them (i.e., lemmatizer, morphological analyzer, dependency parser). The others (i.e., AMR Aligner, Semantic Role Labeler, Named Entity Recognizer) are not ready-to-use services. They should

either be developed from scratch or the existing studies should be recovered/improved. We follow the literature [116] to develop a Named Entity Recognizer (NER) and simply finetune the BERT model over the dataset used in [127]. The following subsections describe the development of the AMR Aligner and the Semantic Role Labeler for Turkish.

5.4.1 An AMR Aligner for morphologically rich and pro-drop languages

Alignment is one of the essential stage of the AMR parsing. It is used to match the concepts of an AMR graph to the words in a sentence. This mapping is then accepted as a reference and used in training concept identifier in the data-driven AMR parser. There are several studies [71,86,92] that rule out the need for AMR alignment. However, these studies consume large amounts of AMR-annotated sentences that are not available in Turkish. In DDP, our parsing approach requires a prior alignment phase for which the proposed aligner is used.

The approaches to automatic AMR alignment widely used in the literature aim at matching concepts (either with fuzzy or semantic matching) with word lemmas using a list of rules. Although these approaches seem to be suitable for English, they do not perform well for languages with different characteristics [100]. MRLs present an interesting challenges for AMR alignment. In MRLs, many concepts emerge from morpho-semantic elements (i.e., suffixes), so that multiple concepts may derive from a single word. An alignment approach based only on word-concept matching [67,79] can not align concepts derived from suffixes, since the word correspondences of such concepts are not explicitly present as words, but is hidden in words as morphemes (e.g., personal markers, modality markers). Figure 5.15 shows an example sentence “Sana geleceğimi bilebilmene şaşırdım” (*I am surprised that you could know that I would be coming to you*) and its AMR graph.

The lexical concepts ‘sen’, ‘gel.01’ (*come*), ‘bil.01’ and ‘şaşır.01’ might be matched by fuzzy matching with the lemma of the words ‘sen’ (*you*), ‘gel’ (*come*), ‘bil’ (*know*) and ‘şaşır’ (*be shocked*). However, the other concepts ‘ben’ (*I*) and mümkün.01 (*possible-01*)) are derived from the personal and modality markers, respectively, and could not be aligned by using fuzzy matching. Extending fuzzy matching with

```
::snt    Sana     geleceğimi                bilebilmene          şaşırdım
::eng    to you  that I would be coming    that you could know  I am surprised
::align  0-1|0.1.1   1-2|0.1.1.0  2-3|0.1.0+0.1  3-4|0+0.0

(s / şaşır.01                                  0
        :ARG0 (b / ben)                        0.0
        :ARG1 (k / mümkün.01                   0.1
               :ARG1 (b2 / bil.01              0.1.0
                      :ARG0 (s / sen)          0.1.1
                      :ARG1 (g / gel.01        0.1.1.0
                             :ARG0 b
                             :ARG4 s))))
```

Figure 5.15 : AMR representation of the sentence "Sana geleceğimi bilebilmene şaşırdım" (*I am surprised that you could know that I would be coming to you*) and its alignment in JAMR format

heuristic rules where the suffixes are given predefined concepts could be a solution, but the following suffixes cannot be matched with it:

- The suffixes whose meaning can be changed depending on the context. For example, the modality marker (*-meli*) can have the meaning of 'obligation' o' 'necessity' depending on the context.
- The suffixes that give rise to predicates that change depending on the root noun. The most prominent example of this group is the very productive suffix *-CI*. It can mean 1) 'a person who sells the object specified in the noun lemma' (e.g., 'simit**çi**' is the person who sells bagels, where 'simit' means bagel), 2) 'a person who runs' the object specified in the noun lemma' (e.g., 'lokanta**cı**' is a person who runs a restaurant, where 'lokanta' means restaurant) 3) 'a person who plays' (e.g., 'bas**cı**' is a person who plays bass guitar, where 'bas' means bass guitar), and so on. To illustrate the examples, see Figure 5.16.

We also believe that a solution based on morphemes (i.e., aligning morphemes using a predefined list) is not a reliable solution since it requires prior morphological analysis. With this solution, an aligner would become very dependent on the performance of the morphological analysis and its errors would propagate throughout the alignment.

We propose an alignment strategy which relies on word-concept similarity and tree traversal. It consists of two steps. In the first step, a map is created on which the

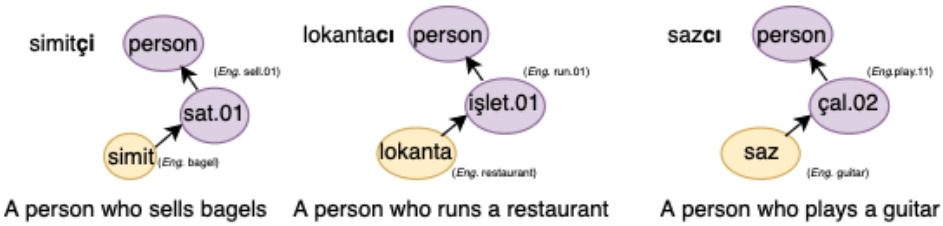

Figure 5.16 : AMR representations of example nouns derived by the very productive suffix *CI*. The purple nodes indicate those created by the suffix; the yellow nodes are the root nouns.

concepts and their word correspondences are paired. This map is created based on the semantic similarity between trained word embeddings for the words of the sentence and the node labels of the graph. The mapping does not necessarily include all concepts. The second step focuses on matching all concepts. First, it starts aligning with the concept-word pairs in the mapping obtained in the first step. Then, the remaining concepts (i.e., morphologically derived and abstract concepts) are matched by traversing the AMR graph through the mapping. For each concept-word pair, the aligner visits the neighbors of the concept by following the heuristically determined paths, and any unaligned neighbors are simply added to the alignments for the word. In the reminder of this subsection, we explain our alignment strategy and give its evaluation.

5.4.1.1 Similarity mapping

Similarity mapping aims to create a map in which concepts have their corresponding words in the sentence. We call each element in the map "concept-word pair" for short. We create this map by using both syntactic and semantic similarities. This approach is similar to TAMR [19], but we do not use morpho-semantic matching. The agglutinative nature of Turkish provides a direct link between nouns invoking verbs and verb frames. Therefore, semantic similarity can be easily used to match such nouns and the use of additional databases is not necessary.

The construction of the map starts with the similarity calculation. For each lemma-concept pair in the cross of lemma and concept, a similarity value is computed. We use Fasttext vectors. The minimum similarity value was empirically set to 0.5 and the lemma-concept pairs with a higher similarity value are considered 'close'. The

closest ones are matched with each other. We accept a pair as a 'the closest pair' if the closest concept of a lemma 'A' is B, while the closest pair of B is A. It should be noted that the aligner allows the lemma to assign more than one concept since there may be cases where A should have more than one concept as a pair.

In some cases, word vectors fail to semantically converge DS derived words even though they share the same root. We take advantage of the fact that the roots are the same and use syntactic similarity[12]. We should point out that we do not use this approach in all cases because the most common light verbs of Turkish contain only two characters (i.e., 'al' (*take* or *get*), 'ol' (*become* or *happen*) and 'et' (*make* or *do*)) and they are syntactically very similar to many words. After these two similarity matching processes, the remaining words that cannot be mapped to any concept are assumed not to contribute to sentence meaning.

Similarity mapping seems , but ellipses make matching difficult. An elliptical construction is the omission of one or more words, which we call omitted words, and whose existence can be understood from the remaining words in the context. The AMR representation of such constructions varies from language to language [39,41]. Similar to [39], the omitted words are also represented with concepts in Turkish AMR. As a result, concepts may appear in the AMR graphs even though their correspondence words do not exist in the sentence. We call these concepts 'elliptic concepts' since they should align with the omitted words. Our approach to align elliptic concepts is to align them with the words that help infer the elided words through semantic inference. For simplicity, we call these words infer words.

We examine the alignment of elliptic concepts in two categories: Alignment with repetitions and alignment with antecedents. The similarity mapping of the first category is straightforward. We can easily align the elliptic concepts (i.e., gapping ellipses) with repetitions, since similarity mapping accomplishes this task without any additional process. In the sentence "Herkes şeker (*verirdi*), o çikolata verirdi." (*Everybody would give chocolate*), he/she would give a candy.), 'verirdi' in the parenthesis is elided, but we can understand its existence through the last predicate

[12]We set a threshold of 0.95 for the similarity score

(i.e., the infer- word). The AMR graph must contain two ‘ver.01’ (*give*) frames since two different people perform different actions. We map both ’ver.01’ concepts to ’verirdi’ and show its alignment in Figure 5.17.

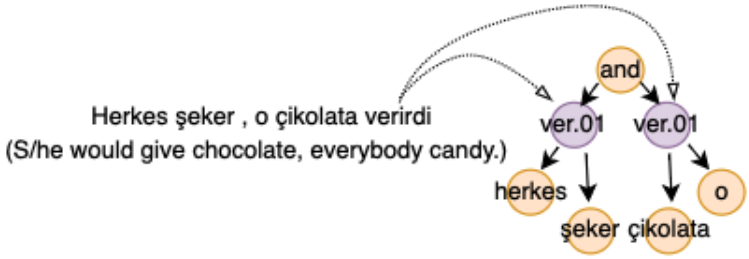

Figure 5.17 : The similarity mapping of a representative sentence of the gapping ellipses (left). The purple colored nodes of the AMR representation (right) indicate the nodes in focus. The dashed lines show the similarity mapping.

The latter category deserves more attention, as the morphologically rich nature of Turkish poses additional challenges. In this category, the meaning of the elided words is carried by the suffixes attached to the antecedents. This means that the elliptic concept should actually be directed to some other words (i.e., antecedents) that are neither semantically nor syntactically similar, even if there are completely identical words in the sentence. The nominal ellipsis is one such example of this type. The elliptical term representing the omitted noun should be aligned with the adjective (e.g., nominal adjectives). Figure 5.18 shows such an example of the alignment of an English sentence (“S/he preferred the red dress over the white.”).

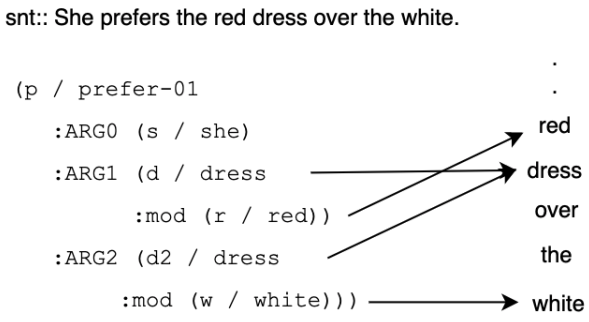

Figure 5.18 : The alignment of a representative sentence of the “She prefers the red dress over the white”. The arrows show the concepts and their corresponding lemmas.

The elliptic concept (i.e., the second dress) is also derived from ‘dress’. However, in Turkish, the meaning of ‘dress’ (first dress) is rendered by the suffix ‘a’ attached to the

adjective. Figure 5.19 shows the alignment. This situation yields the need to map the elliptic concept onto the nominal adjective.

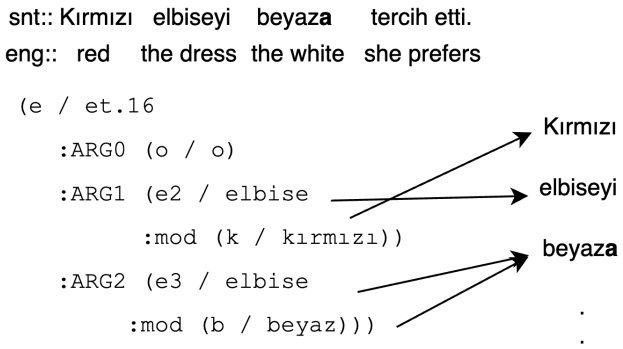

Figure 5.19 : The alignment of a representative sentence of the "Kırmızı elbiseyi beyaza tercih etti". The arrows show the concepts and their corresponding lemmas. The bold letter *-a* attached to 'beyaz' (*white*) represents the dative maker.

To deal with this, we add a disambiguation step. Similarity mapping and disambiguation work simultaneously. If there is more than one concept candidate paired with a single word within the sentence, a disambiguator is invoked to decide whether elimination of a concept-word match is required: At this stage, multiple mapping is allowed for the first category described above (i.e., gapping ellipses); for the second category (i.e., nominal ellipses), the disambiguator selects one of the candidates. It performs this selection by looking for common syntactic structures (i.e., modifiers of the concept-lemma pair in focus) between the candidate concept and the lemma. We generally assume that modifiers (e.g., adjectives describing a noun) would be on the left side of the noun in the actual word order of the sentence. Although this assumption is true in most cases for English and some other languages where modifiers often precede nouns, the direction can be changed as needed for the language in focus. First, the aligner computes an overlap score between the 1-degree neighbors of each candidate and the adjacent words in the 1-word window of the word in focus (i.e., the focus word to be assigned). Then, the candidate with the higher overlap score is matched with the word in focus. Possible assignments of the word in focus 'elbise' (*dress*) in the current sentence in Figure 5.19 are, for example, both concepts e2 and e3. The overlap between the neighbors of the word in focus is the word 'kırmızı' (*red*) shown in Figure 5.20, which eliminates the second possible mapping (the white dress) using our assumption. The similarity mapping is given as Algorithm 1 in Appendix B.

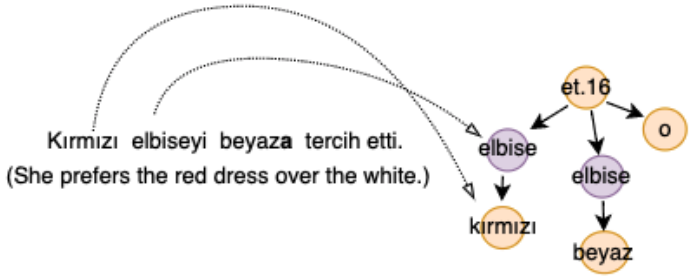

Figure 5.20 : The similarity of a representative clause to the nominal adjective (left). The purple nodes of the AMR representation (right) indicate the nodes in focus. The dashed lines show the similarity mapping.

5.4.1.2 Alignment algorithm

The mapping process begins with a similarity assignment, in which pairs are determined. The details of the mapping process have already been described in the previous section. Algorithm 2 in Appendix B provides the alignment algorithm whose definitions we will use in our explanations. First, we would like to recall the definitions given in above. The input sentence $I = \{w_1, w_2, ..., w_n\}$ where n is the number of words, the AMR graph is $G = \{C, R\}$, where $C = \{c_1, c_2, ..., c_m\}$ is the set of concepts and R is the relation set between these concepts. Note that concept indices and word indices are not directly related. The similarity mapping produces a list, each element of which is also a list (pl). This list pl contains $< w_j, c_i >$ pairs, where j depicts the word order index of the current word within the sentence.

Our aligner processes each $< w_j, c_i >$ pair and first aligns c_i with w_j. Then it searches for c_i's one-edge away neighbors to find mismatched concepts from the previous phase. c_i is accepted as the central node and the aligner visits its neighbors. Suppose the k^{th} neighbor node of c_i is $c_{i_{neigh-k}}$. If $c_{i_{neigh-k}}$ has a pair of words, the aligner returns to c_i. Otherwise, $c_{i_{neigh-k}}$ is added to a list of visited concepts and the aligner goes to $c_{i_{neigh-k}}$ to search for unassigned concepts among the neighbors of $c_{i_{neigh-k}}$. This recursive process ends when there are no more mapped concepts in the neighborhood of c_i, which we define as reachable nodes of c_i. The aligner returns to c_i and the concepts added to the list of visited concepts during the neighborhood search are aligned with w_j. Then the aligner moves to another concept-word pair and repeats the same steps. The alignment

algorithm is finished when all pairs have been processed. Figure 5.21 illustrates the tree traversal phase of the alignment algorithm using the example in Figure 5.15. The colors in the figure are the indicators pointing out the pairs with which nodes are aligned. The dashed arrows show the path that the aligner travels as it performs the alignment of the nodes in the process.

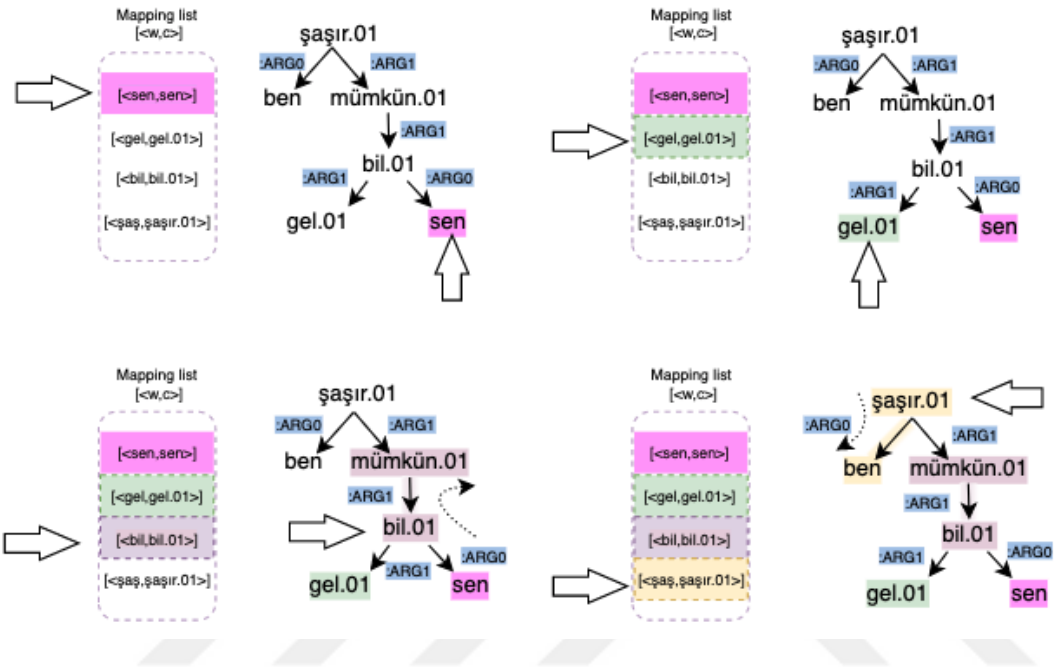

Figure 5.21 : The illustration of the Alignment Algorithm. The similarity mapping list is on the left, where the aligner processes the pairs, and the AMR representation is on the right. The bold white arrows indicate components that are being processed.

The order of concepts in the mapping list is crucial for our aligner, as it greedily searches for adjacent nodes. The unmapped concepts derived from the morphology should be reached first through their children nodes since they tend to appear on top of the lexical concepts in the AMR graph. We add a sort operation that orders the concepts based on their level in the AMR graph. This operation ensures that leaf nodes are processed before top nodes. We also force the aligner to visit neighboring nodes only through the allowed path, so that some nodes can be reached only through certain relations. These constraints guarantee the alignment of the abstract concepts of AMR. For example, the concepts '*-quantity*' concepts are only reachable through the relations *:unit* and *:value*. The restrictions are taken from the JAMR rules, which are responsible for the alignment of abstract concepts. See Table B.1 in the Appendix for

a complete list of allowed relations between concepts. We use a recursive procedure in the alignment algorithm since a word may process nested concepts. As you recall the example 'simitçi' (the person who sells bagels) from Section 5.4.1, the very productive suffix *CI* creates two concepts 'person', 'sell.01', where 'person' is the parent of the frame 'sell.01'. We use the recursive search explained above to find these concepts within the nested relation chains. We also remove reentrancy relations at the beginning of the alignment procedure and convert the graphs into trees. The reasons are that (i) we aim to align morphemes whose alignments are graph fragments; reentrancy connections appear on the linguistic phenomena such as co-reference, coordination, repetition, etc. [128] not on morphology morphological features. Therefore, we assume that the graph fragments do not contain reentrancy connections. (ii) the majority of the reentrancy relations come from the personal suffixes, whose concepts are mostly morphological in origin. Figure 5.22 shows the final alignment of the given example in Figure 5.15. In the figure, it is shown the concept 'ben' comes from the personal suffix *-Im* and can be aligned with 'geleceğ**im**i' or 'şaşırd**ım**', both alignments are correct. Since one of them is sufficient for the aligner to be used in concept generation as the first stage of parsing, we ignore the reentrancy connections during alignment to be handled later during parsing.

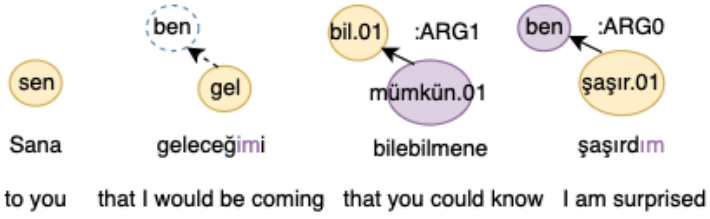

Figure 5.22 : The final alignment generated by the aligner. The purple nodes denote concepts derived from the morphology. We also indicate the 1^{st} personal suffix *-(I)m* with the same color. The dashed concept shows the fact that the concept 'ben' could also be aligned with the word 'geleceğimi'.

Up to this stage, the aligner produces alignments for words. However, to create alignments for word spans (e.g., named entities, reduplications, multi-word expressions), we need an additional step to combine some words and their alignments. Therefore, we add an additional two-stage post-processing step: the first stage focuses on the alignment of named entities: It detects the consecutive words that were initially

aligned to some concepts associated with the same 'name'[13] concept, and these words are merged into word spans while their AMR representations are merged into subgraphs. The second stage focuses on multi-word expressions (i.e., idioms) and reduplications. The algorithm can align only one word of such constructions, leaving the other words of the expression or reduplication unaligned. The postprocessor examines each unaligned word within a sentence and creates word spans by combining them with their consecutive neighbours whose associated concept name contains the lemma of the unaligned word. For languages where these phenomena do not occur, this final post-processing step can be omitted.

5.4.1.3 Experiments and evaluation

We evaluate the performance of the aligner using the Turkish AMR corpus by performing two experiments. The first is the traditional method used in the literature and compares our aligner with the TAMR [79], JAMR [67], and the Portuguese aligner (short, PrAMR) [100]. We use the same evaluation method as for JAMR[14]. e adapt these aligners to Turkish by (1) replacing the predefined dictionaries such as months, conjunctions, etc. with their Turkish equivalents; (2) modifying the source codes of TAMR and PrAMR to use fasttext in TAMR and PrAMR; (3) setting the thresholds to 0.5 and 1.5, respectively. Since PrAMR uses lexical lists for named entities, we localized them as much as possible using a translator from Portuguese to Turkish and added additional Turkish gazetteers. The second experiment evaluates the effectiveness of our approach and examines the alignment performance of our aligner on different concept types. We evaluate our aligner on sentence constituents involving only the concept types in focus.

For the evaluation, 100 sentences are first randomly selected from the corpus. Then, an annotator manually aligns the concepts of the AMR graphs with the words in the sentences in two iterations. In the first iteration, the alignments were created from scratch. In the second iteration, the same annotator checked the correctness of the alignments and corrected the alignments that had errors. Table 5.9 shows the

[13] In AMR, the abstract 'name' concept is used for representing the named entities.
[14] https://github.com/jflanigan/jamr/blob/Semeval-2016/src/EvalSpans.scala

results. Our aligner achieved an F1 score of 87% and outperformed the other aligners developed for English and Portuguese by achieving a relative error reduction of up to 76%. Although TAMR and JAMR have a precision score relatively close to our aligner, they fall far short of the F1 score due to their low recalls. The recalls clearly show that the proposed alignment methods fail for about half of the concepts; these are the concepts derived from morphology. Furthermore, the alignment approach with Word Mover's Distance (PrAMR) has poorer performance than fuzzy matching.

Table 5.9 : The evaluation of our aligner

Output	**P**	**R**	**F1**
JAMR	0.73	0.48	0.58
TAMR	0.70	0.43	0.53
PrAMR	0.55	0.39	0.45
Ours	**0.89**	**0.84**	**0.87**

On the other hand, Table 5.10 shows the alignment performance of the aligner for different concept types. The performance of our aligner parallels the overall result, except for elliptic concepts. The performance for this phenomenon falls behind the overall score.

Table 5.10 : Alignment performance of our aligner on different concept types

	P	**R**	**F1**
Elliptic Concepts	0.60	0.42	0.50
NEs	0.86	0.89	0.88
Abstract Concepts	0.90	0.82	0.86
Morphological Concepts	0.87	0.86	0.86

We make a further error analysis to identify the weaknesses of our aligner. One of the flaws of our aligner is the mismatch of certain concepts. Since our alignment algorithm greedily searches for unassigned concepts, any error in the mapping phase leads to incorrect alignments. Although our aligner leverages the power of pre-trained word embeddings, it fails to match the punctuation marks when they generate concepts, especially when they have a coordination role in the sentence: For example, the comma represents the concept 'and', and the colon represents the concept 'de.01' (*say.01*), however these punctuation marks are not similar to the concept names. The alignment

of light verbs is another case where our aligner fails. In the Turkish Propbank [51] they are represented as frames of auxiliary verbs, as they are in the AMR Therefore, based on semantic similarity, our aligner assigns only the verb part; the first part of the verb is left unmatched. For example 'tercih et-' (*to prefer*) is represented as 'et.16' our aligner aligns only 'et' (*do*). Our aligner also performs poorly in aligning the auxiliary verb 'ol'. This verb has 26 frames, including the widely used meanings 'have' (ol.04) and 'become' (ol.03). When it occurs multiple times in the same sentence, the aligner does not have enough information to distinguish these frames. As a result, incorrect mappings may occur. One way to solve this ambiguity problem might be to include Propbank verb frames as an external resource in future work. One should note that this type of addition would increase the alignment cost.

5.4.2 Neural semantic role labeler for Turkish

Semantic Role Labeling (SRL) is the task of finding the argument structures of verbs in a sentence. It consists of four subtasks: predicate detection, predicate sense disambiguation, argument identification and argument classification. In the first two tasks, the actions in the sentence are found and their 'verb frames' are determined. The Proposition Bank (shortly,PropBank) [7] is used for the meanings of these verb frames in SRL. These frames are represented by numbers and have their own element structure: ARG0, ARG1, ARG2, etc. In general, ARG0 denotes the performer of the action (*PROTO-AGENT*), while ARG1 represents the object directly affected by the action (*PROTO-PATIENT*). For example, in the sentence "Kardeşim bu akşam kar yağışından eve gidemedi" (*My sister/brother could not go home last night*), the word 'gidemedi' is shown with the verb frame git.01 because it carries the action meaning. The last two subtasks (argument identification and argument classification) of the SRL find the words that correspond to the arguments of the verb frames in the sentence and determine their argument type. In the previous example, the word span 'kardeşim' (*my sister/my brother*) is the ARG0 of 'git.01'. There are two main schemes for SRL annotations in the literature: 1) span-based 2) dependency-based. Although PropBank defines arguments as word spans, the Conll 2008 [129] and 2009 [130] shared tasks

adopt this second scheme by annotating the head words of arguments. Figure 5.23 shows both annotation schemes.

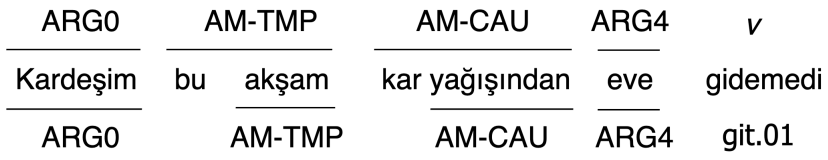

Figure 5.23 : The scheme of SRL annotations is shown in a single sentence. The span-based annotations are placed on the words, while the annotation below the words displays a dependency-based annotation scheme.

SRL, with the semantic information it provides, has an important place for many NLP subtasks (question answering [131], machine translation [132], etc.). AMR also relies heavily on SRL, it uses verb predicates and argument relations in its representations. In this study, we use SRL tags as features in the development of both parsers, resulting in the need for an SRL tool for Turkish. Although semantic task analysis is well researched in the literature, the studies for Turkish are not mature enough. The studies conducted for Turkish SRL have mainly focused on syntactic features, which makes them heavily dependent on additional tools such as dependency parsing and morphological analyzers. Our goal is to annotate the tLP with an SRL tool since the tLP the gold standard annotations (i.e., morphology and dependency annotations) are not available. Therefore, we prefer to develop an SRL tool that adapts a pre-trained language model to Turkish SRL. In the following sections, the proposed SRL model is described and its evaluation is given.

5.4.2.1 Model architecture

We follow the solution proposed in [115], where the SRL task is treated as a sequence and the identification and classification of arguments is solved by a model. We adapt the proposed model architecture to Turkish with a small modification. In the original architecture, a context vector is created for each token by combining the BERT [116] output layer with the vector called 'verb indicator vector'. The context vectors are fed into the long short term memory [133](LSTM) layer and the multi-layer perceptron (MLP) layer respectively, and predication is done using the outputs of the MLP layer. This verb indicator vector helps the model to distinguish between verbs and nonverb

symbols in the sentence during training. Considering this, we consider the verb indicator vector with a different approach and prefer to use this vector as input to the pre-trained model.This approach is the reinterpreted version of the aforementioned study by allennlp [134].

We remove the LSTM later and make the final predictions with the outputs of the MLP layer. At this point, we note that it is possible to use more than one verb in a sentence, so it is necessary to specify the verb whose arguments are to be found in the model. In other words, the sentence and the verb to be predicted should be given as input to the model. In the study we followed, the verb to be predicted is indeed added as a continuation token at the end of the input sentence (input example: [CLS] Kardeşim bu akşam eve gidemedi [SEP] gidemedi [SEP]). However, since we have specified the verb in focus with the verb indicator vector, there is no need to specify it again. Figure 5.24 shows the model architecture we used.

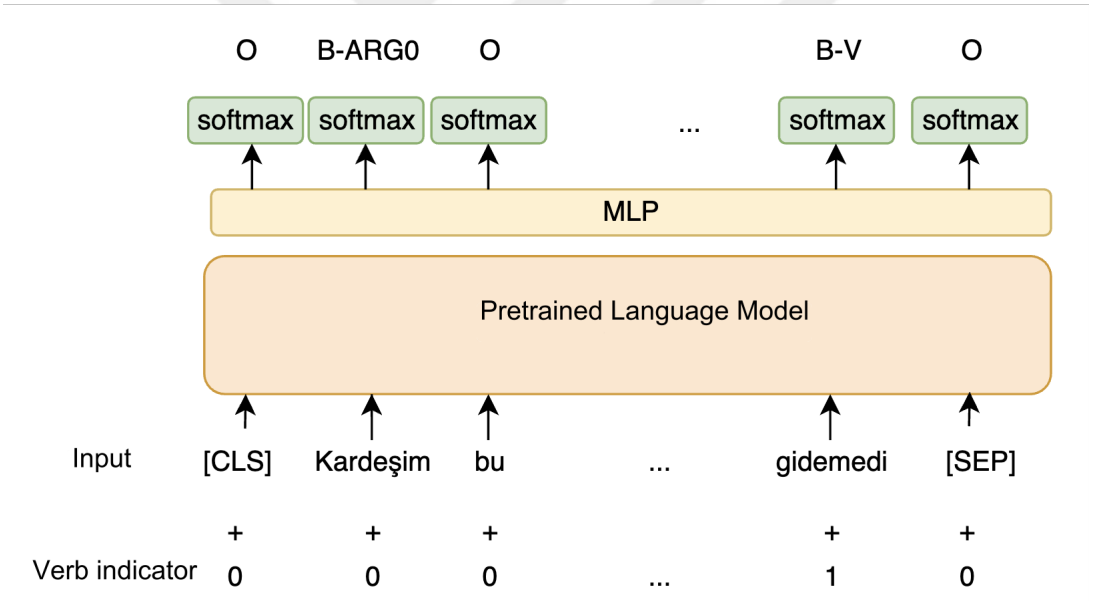

Figure 5.24 : The base model architecture used for SRL

5.4.2.2 Experiment and evaluation

The training, validation, and testing portions of the IMST-UD [61] are used for evaluation. Since in the dataset each derivational group is represented as a separate token, words are separated from their derivational boundaries and their derivational

suffixes are connected to the root via the dependency relation 'DERIV'. We combine these suffixes using the DERIV and use the words in their surface form. Table 5.11 shows the statistics for the training, validation and testing part of our dataset. There are also arguments of causative and reflexive verbs in the ARGx group (within the elements coming from PropBank), and there is an imbalance in the distribution of these arguments. Due to space limitations, we do not report the exact number of these roles; they are gathered under ARGx.

Table 5.11 : Statistics of the SRL datasets used in training, validation, and testing

Arguments	Train	Validation	Test
O	92640	24237	22526
Verb	7946	1867	1800
ARGx	9294	2199	2120
ADV	211	52	0
CAU	232	63	48
COM	55	11	17
DIR	50	13	14
DIS	98	19	20
EXT	272	59	50
GOL	269	64	74
INS	102	27	35
LOC	537	110	74
LVB	471	104	109
MNR	996	253	221
MOD	6	0	0
NEG	49	10	8
TMP	1103	237	252
TWO	74	12	16

We evaluate our model only for the identification and classification of arguments with the goal of finding a parallel to [117]. Since the problem achieves high performance in the first two tasks with simple classifiers, we focus on the last two subtasks, which are relatively difficult problems. We design an experiment to investigate the effect of different language models on the task and the parameters of 5 pre-trained language models (BERTurk (cased, 32k), ELECTRA [135] base (case) (trained on mC4 collection) and ConvBERTurk [136] (case) (trained on mC4 corpus))) [137] are fine-tuned. With this experiment, we also have the opportunity to observe the success of the language modeling architectures used in this study in establishing a semantic

relation between Turkish words. The results of the char-LSTM and morph-LSTM models [117] are used as reference values for this evaluation to compare the context syntax. Table 5.12 shows the obtained results.

Table 5.12 : Results of different language models.

Language Model	Precision	Recall	F1
BERTurk (cased, 32k)	69	70	**70**
ELECTRA Base (cased)	66	70	68
ELECTRA Base mC4 (cased)	66	68	67
ConvBERTurk (cased)	66	65	66
ConvBERTurk mC4 (cased)	65	66	65
char-LSTM	-	-	56
morph-LSTM	-	-	59

The results show that language models score higher than those obtained with syntactic features. The most successful model, BERTurk, increases the scores of the char-LSTM model by 14 points and those of the morph-LSTM by 10 points. It should be noted that the morph-LSTM score obtained with the gold standard dataset does not include errors from the morphological analysis. In real-life use, errors from the morphological analysis will affect the performance. Our approach does not depend on additional resources, so it shows the final results. We can conclude that models that focus on context vectors perform better than those that focus on syntax for SRL. When the pre-trained language models are compared, the BERT architecture can represent the verb-argument relationship better than others, with a small difference. Moreover, it can be observed that the models trained by adding the multilingual mC4 corpus is has poorer performance in understanding the verb-argument relation than the baseline models. Thus, it can be concluded that monolingual models in Turkish are more successful than multilingual models in semantic representation.

When we perform an error analysis, we observe that the model tends to misclassify rather than fail to recognize them. The reasons can be listed as follows: 1) difficulty in distinguishing causative and reflexive structures of the model: These voice structures come from voice suffixes and completely changes the argument structures of the predicates. 2) marking the other verbs with an O, while the sentence-focusing verb is given in the approach with the input sentence: verbal nominal usage is highly common

in Turkish and a sentence may have more than 3 action-represented nominals. Labeling non-focused nominals O leads the model to learn them as O. These conclusions point to the focus of future SRL studies and show that, unlike English, a language model-based approach alone is not sufficient for Turkish SRL. Since SRL is highly related to syntactic features such as PoS tags, and dependency relations, their integration into pretrained deep models seems to be a huge impact. In light of our findings, we believe that it will be beneficial to adapt language models more effectively to the study and use a hybrid model where syntactic features are specified as attributes in future Turkish SRL studies.

6. CONCLUSIONS

Abstract meaning representation is biased toward English, and its adaptation to languages with rich morphology may pose particular design problems. Turkish is a prominent example of morphologically rich languages, and its agglutinative nature makes it necessary to reconsider the Abstract Meaning Representation. For the first time in the literature, we have introduced the Turkish AMR representation framework. It mainly shows our solutions for dealing with the highly productive derivational and inflectional morphology of Turkish. We have presented AMR-oriented definitions for the rich derivational morphology that cannot be used directly in AMR representations as presented in existing knowledge bases or dictionaries. Inflectional morphology is responsible for important functions such as indicating personal markers, modality, negativity, etc., each of which must be mapped to an appropriate AMR component. We have provided this mapping in the context of Turkish AMR. Moreover, this study presents the first-ever AMR corpus for Turkish, which contains 700 AMR-annotated sentences (600 sentences from the Itu-Metu-Sabancı Treebank, 100 from the novel The Little Prince). This corpus and the framework have been created simultaneously following the annotation technique Model-Annotate-Model-Annotate. We have discussed the whole annotation process: modeling with a data-driven approach, annotating, remodeling with a knowledge-driven approach, and updating the annotation. We have evaluated the quality of the annotations by measuring the inter-annotator agreement between annotators with the Smatch score developed for this measure. We have also investigated the agreement when annotating linguistic phenomena in focus. Furthermore, we have introduced rule-based and data-driven AMR parsers for Turkish. The rule-based AMR parser has been developed on top of IMST, where the syntactic (PoS and morphology tags, dependency relations) and semantic annotations (semantic role labels) are available. The rule-based parser has been designed similarly to the transition-based parsers with eight parsing actions. It parses the sentence under the control of the rule list, which contains the action rules

and the mappings that contain the words invoking abstract concepts and the morphemes deriving concepts. We have used this parser to speed up the manual annotation process and demonstrated that using such a tool speeds up the annotation speed. We have evaluated the rule-based parser on the Turkish AMR corpus and it achieves a Smatch score of 60 F1, which is a strong baseline for Turkish AMR parsing. We have also presented a data-driven parser. It is designed as a pipeline of two consecutive steps, each consisting of several classifiers. We use both discrete and continuous features in the development of classifiers. Discrete features are mostly syntactic features such as dependency relations, PoS tags, etc. For continuous features, we have preferred to use Fasttext word vectors. We have trained the data-driven parser on the IMST part of the Turkish AMR corpus and evaluated it on the Little Prince sentences. The parser achieves a Smatch score of 17% F1 and has poorer performance than the rule-based approach, the possible causes of which have already been discussed. Last but not least, we use assistance tools in the development and evaluation of the parsers presented in this study. While some of these tools are available as ready-to-use services, such as the dependency parser, the morphological analyzer, and the PoS tagger, the remaining has been developed from scratch. We introduce the first AMR aligner for morphologically rich and pro-drop languages, capable of matching morphologically derived concepts with sentence words. We have created a gold AMR alignment test set of 100 sentences by manual annotation and evaluated our aligner on this data set. We have achieved significant success by improving the alignment scores of the adaptation of previously proposed aligners into Turkish up to 0.87 F1 scores. In addition, we use pre-trained language models for Turkish semantic role labeling and investigate their impact on this task. BERT has achieved the best performance with 0.7 F1 scores. We provide all annotated resources and source codes of the parser and aligner for further studies We should also point out that AMR is not the only option for semantic representation of Turkish; it is possible to apply alternative representations in the coming years. We believe that our study, which is the first attempt to reveal the fundamental challenges in formal meaning representation of Turkish, will also shed light on these future studies.

REFERENCES

[1] **Abend, O. and Rappoport, A.** (2013). Universal conceptual cognitive annotation (UCCA), *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp.228–238.

[2] **Zhao, J.**, **Xue, N.**, **Van Gysel, J. and Choi, J.D.** (2021). UMR-Writer: A Web Application for Annotating Uniform Meaning Representations, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, pp.160–167, https://aclanthology.org/2021.emnlp-demo.19.

[3] **Cai, S. and Knight, K.** (2013). Smatch: an evaluation metric for semantic feature structures, *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp.748–752.

[4] **Pustejovsky, J.**, **Bunt, H. and Zaenen, A.**, (2017). Designing annotation schemes: From theory to model, Handbook of Linguistic Annotation, Springer, pp.21–72.

[5] **Baker, C.F.**, **Fillmore, C.J. and Lowe, J.B.** (1998). The berkeley framenet project, *COLING 1998 Volume 1: The 17th International Conference on Computational Linguistics*.

[6] **Schuler, K.K.** (2005). *VerbNet: A broad-coverage, comprehensive verb lexicon*, University of Pennsylvania.

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

[8] **Kamp, H. and Reyle, U.** (2013). *From discourse to logic: Introduction to modeltheoretic semantics of natural language, formal logic and discourse representation theory*, volume 42, Springer Science & Business Media.

[9] **Van Gysel, J.E.**, **Vigus, M.**, **Chun, J.**, **Lai, K.**, **Moeller, S.**, **Yao, J.**, **O'Gorman, T.**, **Cowell, A.**, **Croft, W.**, **Huang, C.R.** *et al.* (2021). Designing a uniform meaning representation for natural language processing, *KI-Künstliche Intelligenz*, *35*(3), 343–360.

[10] **Banarescu, L.**, **Bonial, C.**, **Cai, S.**, **Georgescu, M.**, **Griffitt, K.**, **Hermjakob, U.**, **Knight, K.**, **Koehn, P.**, **Palmer, M. and Schneider, N.** (2013). Abstract

meaning representation for sembanking, *Proceedings of the 7th linguistic annotation workshop and interoperability with discourse*, pp.178–186.

[11] **Kamp, H.** (1988). Discourse Representation Theory: What it is and where it ought to go., *Natural Language at the computer*, *320*(1), 84–111.

[12] **Bos, J.**, **Basile, V.**, **Evang, K.**, **Venhuizen, N.J. and Bjerva, J.**, (2017). The groningen meaning bank, Handbook of linguistic annotation, Springer, pp.463–496.

[13] **Kasper, R.T.** (1989). A flexible interface for linking applications to Penman's sentence generator, *Speech and Natural Language: Proceedings of a Workshop Held at Philadelphia, Pennsylvania, February 21-23, 1989*.

[14] **Weischedel, R.**, **Hovy, E.**, **Marcus, M.**, **Palmer, M.**, **Belvin, R.**, **Pradhan, S.**, **Ramshaw, L. and Xue, N.** (2011). OntoNotes: A large training corpus for enhanced processing, *Handbook of Natural Language Processing and Machine Translation. Springer*, 59.

[15] **Oral, E.**, **Acar, A. and Eryiğit, G.** (2022). Abstract meaning representation of Turkish, *Natural Language Engineering*, 1–30.

[16] **Bos, J.** (2016). Squib: Expressive Power of Abstract Meaning Representations, *Computational Linguistics*, *42*, 527–535.

[17] **Žabokrtský, Z.**, **Zeman, D. and Ševčíková, M.** (2020). Sentence Meaning Representations Across Languages: What Can We Learn from Existing Frameworks?, *Computational Linguistics*, *46*(3), 605–665.

[18] **Dohare, S. and Karnick, H.** (2017). Text Summarization using Abstract Meaning Representation.

[19] **Liu, F.**, **Flanigan, J.**, **Thomson, S.**, **Sadeh, N. and Smith, N.A.** (2018). Toward abstractive summarization using semantic representations, *arXiv preprint arXiv:1805.10399*.

[20] **Liao, K.**, **Lebanoff, L. and Liu, F.** (2018). *Abstract Meaning Representation for Multi-Document Summarization*.

[21] **Song, L.**, **Zhang, Y.**, **Peng, X.**, **Wang, Z. and Gildea, D.** (2016). AMR-to-text generation as a traveling salesman problem, *arXiv preprint arXiv:1609.07451*.

[22] **Song, L.**, **Zhang, Y.**, **Wang, Z. and Gildea, D.** (2018). A graph-to-sequence model for AMR-to-text generation, *arXiv preprint arXiv:1805.02473*.

[23] **Damonte, M. and Cohen, S.B.** (2019). Structural neural encoders for AMR-to-text generation, *arXiv preprint arXiv:1903.11410*.

[24] **Wang, T.**, **Wan, X. and Jin, H.** (2020). AMR-To-Text Generation with Graph Transformer, *Transactions of the Association for Computational Linguistics*, *8*, 19–33.

[25] **Mager, M.**, **Astudillo, R.F.**, **Naseem, T.**, **Sultan, M.A.**, **Lee, Y.S.**, **Florian, R. and Roukos, S.** (2020). GPT-too: A language-model-first approach for AMR-to-text generation, *arXiv preprint arXiv:2005.09123*.

[26] **Zhao, Y.**, **Chen, L.**, **Chen, Z.**, **Cao, R.**, **Zhu, S. and Yu, K.** (2020). Line Graph Enhanced AMR-to-Text Generation with Mix-Order Graph Attention Networks, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, Online, pp.732–741.

[27] **Fan, A. and Gardent, C.** (2020). Multilingual AMR-to-Text Generation, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, Online, pp.2889–2901.

[28] **Wang, T.**, **Wan, X. and Yao, S.** (2020). Better AMR-To-Text Generation with Graph Structure Reconstruction, ***C. Bessiere**, editor, Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, International Joint Conferences on Artificial Intelligence Organization, pp.3919–3925, main track.

[29] **Bai, X.**, **Song, L. and Zhang, Y.** (2020). Online Back-Parsing for AMR-to-Text Generation, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, Online, pp.1206–1219.

[30] **Jin, L. and Gildea, D.** (2020). Generalized Shortest-Paths Encoders for AMR-to-Text Generation, *Proceedings of the 28th International Conference on Computational Linguistics*, International Committee on Computational Linguistics, Barcelona, Spain (Online), pp.2004–2013.

[31] **Song, L.**, **Gildea, D.**, **Zhang, Y.**, **Wang, Z. and Su, J.** (2019). Semantic Neural Machine Translation Using AMR, *Transactions of the Association for Computational Linguistics*, *7*, 19–31.

[32] **Takase, S.**, **Suzuki, J.**, **Okazaki, N.**, **Hirao, T. and Nagata, M.** (2016). Neural Headline Generation on Abstract Meaning Representation, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Austin, Texas, pp.1054–1059.

[33] **Huang, L.**, **Cassidy, T.**, **Feng, X.**, **Ji, H.**, **Voss, C.R.**, **Han, J. and Sil, A.** (2016). Liberal Event Extraction and Event Schema Induction, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, Berlin, Germany, pp.258–268.

[34] **Li, M.**, **Zareian, A.**, **Zeng, Q.**, **Whitehead, S.**, **Lu, D.**, **Ji, H. and Chang, S.F.** (2020). Cross-media Structured Common Space for Multimedia Event Extraction, *arXiv preprint arXiv:2005.02472*.

[35] **Bonial, C.**, **Donatelli, L.**, **Abrams, M.**, **Lukin, S.M.**, **Tratz, S.**, **Marge, M.**, **Artstein, R.**, **Traum, D. and Voss, C.** (2020). Dialogue-AMR: Abstract Meaning Representation for Dialogue, *Proceedings of the 12th Language Resources and Evaluation Conference*, European Language Resources Association, Marseille, France, pp.684–695.

[36] **Bonn, J.**, **Palmer, M.**, **Cai, Z. and Wright-Bettner, K.** (2020). Spatial AMR: Expanded Spatial Annotation in the Context of a Grounded Minecraft Corpus, *Proceedings of the 12th Language Resources and Evaluation Conference*, European Language Resources Association, Marseille, France, pp.4883–4892.

[37] **Li, B.**, **Wen, Y.**, **Qu, W.**, **Bu, L. and Xue, N.** (2016). Annotating the Little Prince with Chinese AMRs, *Proceedings of the 10th Linguistic Annotation Workshop held in conjunction with ACL 2016 (LAW-X 2016)*, Association for Computational Linguistics, Berlin, Germany, pp.7–15.

[38] **Li, B.**, **Wen, Y.**, **Song, L.**, **Qu, W. and Xue, N.** (2019). Building a Chinese AMR Bank with Concept and Relation Alignments, *Linguistic Issues in Language Technology, Volume 18, 2019 - Exploiting Parsed Corpora: Applications in Research, Pedagogy, and Processing*, CSLI Publications.

[39] **Liu, Y.**, **Li, B.**, **Yan, P.**, **Song, L. and Qu, W.** (2019). Ellipsis in Chinese AMR Corpus, *Proceedings of the First International Workshop on Designing Meaning Representations*, pp.92–99.

[40] **Xue, N.**, **Xia, F.**, **Chiou, F.D. and Palmer, M.** (2005). The Penn Chinese TreeBank: Phrase structure annotation of a large corpus, *Natural Language Engineering*, *11*, 207–238.

[41] **Migueles-Abraira, N.**, **Agerri, R. and Diaz de Ilarraza, A.** (2018). Annotating Abstract Meaning Representations for Spanish, *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC)*, European Language Resources Association (ELRA), Miyazaki, Japan.

[42] **Anchiêta, R. and Pardo, T.** (2018). Towards AMR-BR: A SemBank for Brazilian Portuguese Language, *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC)*, Miyazaki, Japan.

[43] **Sobrevilla Cabezudo, M.A. and Pardo, T.** (2019). Towards a General Abstract Meaning Representation Corpus for Brazilian Portuguese, *Proceedings of the 13th Linguistic Annotation Workshop*, Association for Computational Linguistics, Florence, Italy, pp.236–244, `https://aclanthology.org/W19-4028`.

[44] **Choe, H.**, **Han, J.**, **Park, H. and Kim, H.** (2019). Copula and Case-Stacking Annotations for Korean AMR, *Proceedings of the First International Workshop on Designing Meaning Representations*, pp.128–135.

[45] **Choe, H.**, **Han, J.**, **Park, H.**, **Oh, T.H. and Kim, H.** (2020). Building Korean Abstract Meaning Representation Corpus, *Proceedings of the Second International Workshop on Designing Meaning Representations*, Association for Computational Linguistics, Barcelona Spain (online), pp.21–29.

[46] **Choe, H.**, **Han, J.**, **Park, H.**, **Oh, T.**, **Park, S. and Kim, H.** (2019). Korean Abstract Meaning Representation (AMR) Guidelines for Graph-structured Representations of Sentence Meaning, *Proceedings of the 31th Annual Conference on Human and Cognitive Language Technology*, pp.252–257.

[47] **Xue, N.**, **Bojar, O.**, **Hajič, J.**, **Palmer, M.**, **Urešová, Z. and Zhang, X.** (2014). Not an Interlingua, But Close: Comparison of English AMRs to Chinese and Czech, *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC)*, European Language Resources Association (ELRA), Reykjavik, Iceland, pp.1765–1772.

[48] **Linh, H. and Nguyen, H.** (2019). A Case Study on Meaning Representation for Vietnamese, *Proceedings of the First International Workshop on Designing Meaning Representations*, pp.148–153.

[49] **Feng, L.** (2021). *WISeN: Widely Interpretable Semantic Network for Richer Meaning Representation*, Emory University, Atlanta, GA, undergraduate Honors Thesis, Emory University, Atlanta, GA, 2021.

[50] **Zhu, H.**, **Li, Y. and Chiticariu, L.** (2019). Towards Universal Semantic Representation, *Proceedings of the First International Workshop on Designing Meaning Representations*, Association for Computational Linguistics, Florence, Italy, pp.177–181.

[51] **Şahin, G.G.** (2016). Framing of verbs for Turkish propbank, *Turkish Computational Linguistics*, 3–9.

[52] **Şahin, G.G. and Adalı, E.** (2018). Annotation of semantic roles for the Turkish Proposition Bank, *Language Resources and Evaluation*, *52*, 673–706.

[53] **Choi, J.D.**, **Bonial, C. and Palmer, M.** (2010). Propbank frameset annotation guidelines using a dedicated editor, cornerstone, *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*.

[54] **Babko-Malaya, O.** (2005). Guidelines for Propbank framers, *Unpublished manual, September*.

[55] **Göksel, A. and Kerslake, C.** (2004). *Turkish: A comprehensive grammar*, Routledge.

[56] **Meyers, A.**, **Reeves, R.**, **Macleod, C.**, **Szekely, R.**, **Zielinska, V.**, **Young, B. and Grishman, R.** (2004). The NomBank Project: An Interim Report, *Proceedings of the Workshop Frontiers in Corpus Annotation at*

HLT-NAACL 2004, Association for Computational Linguistics, Boston, Massachusetts, USA, pp.24–31.

[57] **Pustejovsky, J. and Stubbs, A.** (2012). *Natural Language Annotation for Machine Learning: A guide to corpus-building for applications*, " O'Reilly Media, Inc.".

[58] **Bunt, H.** (2015). On the principles of semantic annotation, *Proceedings of the 11th Joint ACL-ISO Workshop on Interoperable Semantic Annotation (ISA-11)*.

[59] **Hermjakob, U.** (2013). AMR editor: A tool to build Abstract Meaning Representations, **Technical Report**, Technical report, ISI.

[60] **Azin, Z. and Eryiğit, G.** (2019). Towards Turkish Abstract Meaning Representation, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, Association for Computational Linguistics, Florence, Italy, pp.43–47.

[61] **Sulubacak, U.**, **Eryiğit, G. and Pamay, T.** (2016). IMST: A Revisited Turkish Dependency Treebank, ***B. Karaoğlan***, ***T. Kışla and S. Kumova***, *editors, Proceedings of TurCLing 2016, the 1st International Conference on Turkic Computational Linguistics*, EGE UNIVERSITY PRESS, Turkey, pp.1–6.

[62] **Buchholz, S. and Marsi, E.** (2006). CoNLL-X Shared Task on Multilingual Dependency Parsing, *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*, Association for Computational Linguistics, New York City, pp.149–164, `https://aclanthology.org/W06-2920`.

[63] **Oral, K.E. and Eryiğit, G.** (2022). AMR Alignment for Morphologically-rich and Pro-drop Languages, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, Association for Computational Linguistics, Dublin, Ireland, pp.143–152, `https://aclanthology.org/2022.acl-srw.13`.

[64] **Oral, E. and Eryiğit, G.** (2022). The Impact of Pre-trained Language Models on Turkish Semantic Role Labelling, *2022 30th Signal Processing and Communications Applications Conference (SIU)*, pp.1–4.

[65] **Oepen, S.**, **Abend, O.**, **Hajic, J.**, **Hershcovich, D.**, **Kuhlmann, M.**, **O'Gorman, T.**, **Xue, N.**, **Chun, J.**, **Straka, M. and Uresova, Z.** (2019). MRP 2019: Cross-Framework Meaning Representation Parsing, *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, Association for Computational Linguistics, Hong Kong, pp.1–27, `https://aclanthology.org/K19-2001`.

[66] **Oepen, S.**, **Abend, O.**, **Abzianidze, L.**, **Bos, J.**, **Hajič, J.**, **Hershcovich, D.**, **Li, B.**, **O'Gorman, T.**, **Xue, N. and Zeman, D.**, editors, (2020). Proceedings of the CoNLL 2020 Shared Task: Cross-Framework Meaning Representation

Parsing, Association for Computational Linguistics, Online, https://aclanthology.org/2020.conll-shared.0.

[67] **Flanigan, J.**, **Thomson, S.**, **Carbonell, J.**, **Dyer, C. and Smith, N.A.** (2014). A Discriminative Graph-Based Parser for the Abstract Meaning Representation, *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, Baltimore, Maryland, pp.1426–1436.

[68] **Flanigan, J.**, **Dyer, C.**, **Smith, N.A. and Carbonell, J.** (2016). CMU at SemEval-2016 Task 8: Graph-based AMR Parsing with Infinite Ramp Loss, *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, Association for Computational Linguistics, San Diego, California, pp.1202–1206.

[69] **Werling, K.**, **Angeli, G. and Manning, C.** (2015). Robust subgraph generation improves Abstract Meaning Representation parsing, *arXiv preprint arXiv:1506.03139*.

[70] **Foland, W. and Martin, J.H.** (2017). Abstract Meaning Representation Parsing using LSTM Recurrent Neural Networks, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, Vancouver, Canada, pp.463–472.

[71] **Lyu, C. and Titov, I.** (2018). AMR parsing as graph prediction with latent alignment, *arXiv preprint arXiv:1805.05286*.

[72] **Zhang, S.**, **Ma, X.**, **Duh, K. and Van Durme, B.** (2019). AMR parsing as sequence-to-graph transduction, *arXiv preprint arXiv:1905.08704*.

[73] **Bai, X.**, **Chen, Y. and Zhang, Y.** (2022). Graph Pre-training for AMR Parsing and Generation, *arXiv preprint arXiv:2203.07836*.

[74] **Wang, C.**, **Xue, N. and Pradhan, S.** (2015). Boosting Transition-based AMR Parsing with Refined Actions and Auxiliary Analyzers, *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, Association for Computational Linguistics, Beijing, China, pp.857–862.

[75] **Damonte, M.**, **Cohen, S.B. and Satta, G.** (2017). An Incremental Parser for Abstract Meaning Representation, *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, Association for Computational Linguistics, Valencia, Spain, pp.536–546.

[76] **Ballesteros, M. and Al-Onaizan, Y.** (2017). AMR parsing using stack-LSTMs, *arXiv preprint arXiv:1707.07755*.

[77] **Wang, C. and Xue, N.** (2017). Getting the Most out of AMR Parsing, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Copenhagen, Denmark, pp.1257–1268.

[78] **Guo, Z. and Lu, W.** (2018). Better Transition-Based AMR Parsing with a Refined Search Space, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Brussels, Belgium, pp.1712–1722.

[79] **Liu, Y.**, **Che, W.**, **Zheng, B.**, **Qin, B. and Liu, T.** (2018). An AMR aligner tuned by transition-based parser, *arXiv preprint arXiv:1810.03541*.

[80] **Peng, X.**, **Gildea, D. and Satta, G.** (2018). AMR Parsing With Cache Transition Systems., *AAAI*, pp.4897–4904.

[81] **Naseem, T.**, **Shah, A.**, **Wan, H.**, **Florian, R.**, **Roukos, S. and Ballesteros, M.** (2019). Rewarding Smatch: Transition-based AMR parsing with reinforcement learning, *arXiv preprint arXiv:1905.13370*.

[82] **Astudillo, R.F.**, **Ballesteros, M.**, **Naseem, T.**, **Blodgett, A. and Florian, R.** (2020). Transition-based Parsing with Stack-Transformers, *arXiv preprint arXiv:2010.10669*.

[83] **Lee, Y.S.**, **Astudillo, R.F.**, **Hoang, T.L.**, **Naseem, T.**, **Florian, R. and Roukos, S.** (2021). Maximum Bayes Smatch Ensemble Distillation for AMR Parsing, *arXiv preprint arXiv:2112.07790*.

[84] **Zhou, J.**, **Naseem, T.**, **Astudillo, R.F.**, **Lee, Y.S.**, **Florian, R. and Roukos, S.** (2021). Structure-aware Fine-tuning of Sequence-to-sequence Transformers for Transition-based AMR Parsing, *arXiv preprint arXiv:2110.15534*.

[85] **Barzdins, G. and Gosko, D.** (2016). Riga at semeval-2016 task 8: Impact of smatch extensions and character-level neural translation on AMR parsing accuracy, *arXiv preprint arXiv:1604.01278*.

[86] **Konstas, I.**, **Iyer, S.**, **Yatskar, M.**, **Choi, Y. and Zettlemoyer, L.** (2017). Neural AMR: Sequence-to-sequence models for parsing and generation, *arXiv preprint arXiv:1704.08381*.

[87] **Van Noord, R. and Bos, J.** (2017). Neural semantic parsing by character-based translation: Experiments with abstract meaning representations, *arXiv preprint arXiv:1705.09980*.

[88] **Xu, D.**, **Li, J.**, **Zhu, M.**, **Zhang, M. and Zhou, G.** (2020). Improving AMR Parsing with Sequence-to-Sequence Pre-training, *arXiv preprint arXiv:2010.01771*.

[89] **Blloshmi, R.**, **Tripodi, R. and Navigli, R.** (2020). XL-AMR: Enabling Cross-Lingual AMR Parsing with Transfer Learning Techniques, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language*

Processing (EMNLP), Association for Computational Linguistics, Online, pp.2487–2500.

[90] **Bevilacqua, M.**, **Blloshmi, R. and Navigli, R.** (2021). One SPRING to rule them both: Symmetric AMR semantic parsing and generation without a complex pipeline, *Proceedings of AAAI*.

[91] **Xia, Q.**, **Li, Z.**, **Wang, R. and Zhang, M.** (2021). Stacked AMR Parsing with Silver Data, *Findings of the Association for Computational Linguistics: EMNLP 2021*, Association for Computational Linguistics, Punta Cana, Dominican Republic, pp.4729–4738, `https://aclanthology.org/2021.findings-emnlp.406`.

[92] **Zhang, S.**, **Ma, X.**, **Duh, K. and Van Durme, B.** (2019). Broad-Coverage Semantic Parsing as Transduction, Association for Computational Linguistics, Hong Kong, China, pp.3786–3798.

[93] **Cai, D. and Lam, W.** (2020). AMR parsing via graph-sequence iterative inference, *arXiv preprint arXiv:2004.05572*.

[94] **Damonte, M. and Cohen, S.B.** (2018). Cross-Lingual Abstract Meaning Representation Parsing, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, Association for Computational Linguistics, New Orleans, Louisiana, pp.1146–1155.

[95] **Uhrig, S.**, **Garcia, Y.R.**, **Opitz, J. and Frank, A.** (2021). Translate, then parse! A strong baseline for cross-lingual AMR parsing, *arXiv preprint arXiv:2106.04565*.

[96] **Anchiêta, R.T. and Pardo, T.A.S.** (2018). A Rule-Based AMR Parser for Portuguese, ***G.R. Simari, E. Fermé, F. Gutiérrez Segura and J.A. Rodríguez Melquiades**, editors, Advances in Artificial Intelligence - IBERAMIA 2018*, Springer International Publishing, Cham, pp.341–353.

[97] **Fellbaum, C.**, **Osherson, A. and Clark, P.E.** (2007). Putting semantics into WordNet's" morphosemantic" links, *Language and technology conference*, Springer, pp.350–358.

[98] **Pourdamghani, N.**, **Gao, Y.**, **Hermjakob, U. and Knight, K.** (2014). Aligning english strings with abstract meaning representation graphs, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp.425–429.

[99] **Brown, P.F.**, **Della Pietra, S.A.**, **Della Pietra, V.J. and Mercer, R.L.** (1993). The mathematics of statistical machine translation: Parameter estimation, *Computational linguistics*, *19*(2), 263–311.

[100] **Anchiêta, R. and Pardo, T.** (2020). Semantically Inspired AMR Alignment for the Portuguese Language, *Proceedings of the 2020 Conference*

on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Online, pp.1595–1600, https://aclanthology.org/2020.emnlp-main.123.

[101] **Kusner, M.**, **Sun, Y.**, **Kolkin, N. and Weinberger, K.** (2015). From word embeddings to document distances, *International conference on machine learning*, PMLR, pp.957–966.

[102] **Pradhan, S.**, **Ward, W.**, **Hacioglu, K.**, **Martin, J.H. and Jurafsky, D.** (2005). Semantic role labeling using different syntactic views, *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pp.581–588.

[103] **Punyakanok, V.**, **Roth, D. and Yih, W.t.** (2008). The importance of syntactic parsing and inference in semantic role labeling, *Computational Linguistics*, *34*(2), 257–287.

[104] **Strubell, E.**, **Verga, P.**, **Andor, D.**, **Weiss, D. and McCallum, A.** (2018). Linguistically-informed self-attention for semantic role labeling, *arXiv preprint arXiv:1804.08199*.

[105] **Roth, M. and Lapata, M.** (2016). Neural semantic role labeling with dependency path embeddings, *arXiv preprint arXiv:1605.07515*.

[106] **FitzGerald, N.**, **Täckström, O.**, **Ganchev, K. and Das, D.** (2015). Semantic role labeling with neural network factors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp.960–970.

[107] **Marcheggiani, D. and Titov, I.** (2017). Encoding sentences with graph convolutional networks for semantic role labeling, *arXiv preprint arXiv:1703.04826*.

[108] **Li, Z.**, **He, S.**, **Cai, J.**, **Zhang, Z.**, **Zhao, H.**, **Liu, G.**, **Li, L. and Si, L.** (2018). A unified syntax-aware framework for semantic role labeling, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp.2401–2411.

[109] **He, S.**, **Li, Z.**, **Zhao, H. and Bai, H.** (2018). Syntax for semantic role labeling, to be, or not to be, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp.2061–2071.

[110] **Zhou, J. and Xu, W.** (2015). End-to-end learning of semantic role labeling using recurrent neural networks, *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)*, pp.1127–1137.

[111] **Tan, Z.**, **Wang, M.**, **Xie, J.**, **Chen, Y. and Shi, X.** (2018). Deep semantic role labeling with self-attention, *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

[112] **Marcheggiani, D.**, **Frolov, A. and Titov, I.** (2017). A simple and accurate syntax-agnostic neural model for dependency-based semantic role labeling, *arXiv preprint arXiv:1701.02593*.

[113] **Cai, J.**, **He, S.**, **Li, Z. and Zhao, H.** (2018). A Full End-to-End Semantic Role Labeler, Syntactic-agnostic Over Syntactic-aware?, *Proceedings of the 27th International Conference on Computational Linguistics*, pp.2753–2765.

[114] **Li, Z.**, **He, S.**, **Zhao, H.**, **Zhang, Y.**, **Zhang, Z.**, **Zhou, X. and Zhou, X.** (2019). Dependency or span, end-to-end uniform semantic role labeling, *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp.6730–6737.

[115] **Shi, P. and Lin, J.** (2019). Simple bert models for relation extraction and semantic role labeling, *arXiv preprint arXiv:1904.05255*.

[116] **Devlin, J.**, **Chang, M.W.**, **Lee, K. and Toutanova, K.** (2018). Bert: Pre-training of deep bidirectional transformers for language understanding, *arXiv preprint arXiv:1810.04805*.

[117] **Şahin, G.G. and Steedman, M.** (2018). Character-Level Models versus Morphology in Semantic Role Labeling, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, Melbourne, Australia, pp.386–396, https://aclanthology.org/P18-1036.

[118] **Wang, C.**, **Xue, N. and Pradhan, S.** (2015). A Transition-based Algorithm for AMR Parsing, *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Association for Computational Linguistics, Denver, Colorado, pp.366–375.

[119] **Grave, E.**, **Bojanowski, P.**, **Gupta, P.**, **Joulin, A. and Mikolov, T.** (2018). Learning Word Vectors for 157 Languages, *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

[120] **Şahin, G.G.** (2022). To Augment or Not to Augment? A Comparative Study on Text Augmentation Techniques for Low-Resource NLP, *Computational Linguistics*, *48*(1), 5–42, https://doi.org/10.1162/coli_a_00425, https://direct.mit.edu/coli/article-pdf/48/1/5/2006622/coli_a_00425.pdf.

[121] **Chawla, N.V.**, **Bowyer, K.W.**, **Hall, L.O. and Kegelmeyer, W.P.** (2002). SMOTE: synthetic minority over-sampling technique, *Journal of artificial intelligence research*, *16*, 321–357.

[122] **Ke, G.**, **Meng, Q.**, **Finley, T.**, **Wang, T.**, **Chen, W.**, **Ma, W.**, **Ye, Q. and Liu, T.Y.** (2017). Lightgbm: A highly efficient gradient boosting decision tree, *Advances in neural information processing systems*, *30*.

[123] **Wright, R.E.** (1995). Logistic regression.

[124] **Chen, T.**, **He, T.**, **Benesty, M.**, **Khotilovich, V.**, **Tang, Y.**, **Cho, H.**, **Chen, K.** *et al.* (2015). Xgboost: extreme gradient boosting, *R package version 0.4-2*, *1*(4), 1–4.

[125] **Quinlan, J.R.** (1996). Learning decision tree classifiers, *ACM Computing Surveys (CSUR)*, *28*(1), 71–72.

[126] **Eryiğit, G.** (2014). ITU Turkish NLP Web Service, *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, Association for Computational Linguistics, Gothenburg, Sweden, pp.1–4.

[127] **Şeker, G.A. and Eryiğit, G.** (2012). Initial explorations on using CRFs for Turkish named entity recognition, *Proceedings of COLING 2012*, pp.2459–2474.

[128] **Blodgett, A. and Schneider, N.** (2021). Probabilistic, Structure-Aware Algorithms for Improved Variety, Accuracy, and Coverage of AMR Alignments, *arXiv preprint arXiv:2106.06002*.

[129] **Surdeanu, M.**, **Johansson, R.**, **Meyers, A.**, **Màrquez, L. and Nivre, J.** (2008). The CoNLL 2008 shared task on joint parsing of syntactic and semantic dependencies, *CoNLL 2008: Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pp.159–177.

[130] **Hajic, J.**, **Ciaramita, M.**, **Johansson, R.**, **Kawahara, D.**, **Martí, M.A.**, **Màrquez, L.**, **Meyers, A.**, **Nivre, J.**, **Padó, S.**, **Štěpánek, J.** *et al.* (2009). The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages, *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, pp.1–18.

[131] **Yih, W.t.**, **Richardson, M.**, **Meek, C.**, **Chang, M.W. and Suh, J.** (2016). The value of semantic parse labeling for knowledge base question answering, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp.201–206.

[132] **Shi, C.**, **Liu, S.**, **Ren, S.**, **Feng, S.**, **Li, M.**, **Zhou, M.**, **Sun, X. and Wang, H.** (2016). Knowledge-based semantic embedding for machine translation, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp.2245–2254.

[133] **Hochreiter, S. and Schmidhuber, J.** (1997). Long short-term memory, *Neural computation*, *9*(8), 1735–1780.

[134] **Gardner, M.**, **Grus, J.**, **Neumann, M.**, **Tafjord, O.**, **Dasigi, P.**, **Liu, N.**, **Peters, M.**, **Schmitz, M. and Zettlemoyer, L.** (2018). Allennlp: A deep semantic natural language processing platform, *arXiv preprint arXiv:1803.07640*.

[135] **Clark, K.**, **Luong, M.T.**, **Le, Q.V. and Manning, C.D.** (2020). Electra: Pre-training text encoders as discriminators rather than generators, *arXiv preprint arXiv:2003.10555*.

[136] **Jiang, Z.H.**, **Yu, W.**, **Zhou, D.**, **Chen, Y.**, **Feng, J. and Yan, S.** (2020). Convbert: Improving bert with span-based dynamic convolution, *Advances in Neural Information Processing Systems*, *33*, 12837–12848.

[137] **Schweter, S.** (2020). *BERTurk - BERT models for Turkish*, `https://doi.org/10.5281/zenodo.3770924`.

APPENDICES

APPENDIX A : Features used in the development of data driven parser
APPENDIX B : The Alignment Algorithm

APPENDIX A: Features used in the development of data driven parser

Table A.1 : The table of Features used in Concept Identification

Feature Nane	**Feature Category**	**Definition**
Word	Lexical	Lemma of word
SurfaceEmb	Embedding	Embedding vector of word's surface
LemmaEmb	Embedding	Embedding vector of word's lemma
Postag	Syntactic	POS tag of word
PersMark	Morphologic	Personal Marker of word
PossessMark	Morphologic	Possessive Marker of word
CaseMark	Morphologic	Case Marker of word
Pronoun	Syntactic	Whether word is pronoun or not
TenseMark	Morphologic	Tense Markers of word
VoiceMark	Morphologic	Voice Marker of word
DepHeadLemma	Syntactic	Lemma of word's dependency head
DepHeadLemmaEmb	Embedding	Embedding of word's dependency head lemma
DepSibs	Syntactic	Set of dependency relations of word's head
DepChild	Syntactic	Set of dependency relations of word's children
SrlArgSibs	Semantic	Set of SRL relations of word's head
SrlRel	Semantic	SRL relation of word
DepRel	Syntactic	dependency relation of word
DepHeadPos	Syntactic	POS tag of word's dependency head
SrlVerbSense	Semantic	Word's verb sense
DepHeadPersMark	Morphologic	Personal Marker of word's dependency head
DepHeadPosessMark	Morphologic	Possessive Marker of word's dependency head
DepHeadCaseMark	Morphologic	Case Marker of word's dependency head
DepHeadPronoun	Morphologic	Whether word's dependency head is pronoun or not
DepHeadTenseMark	Morphologic	Tense Marker of word's dependency head
DepHeadVoiceMark	Morphologic	Voice Marker of word's dependency head
VeryProdCI	Morpho-Semantic	Whether words have the suffix *-CI*
sIz/lI	Morpho-Semantic	Whether words have the suffix *-sIz/lI*
NextWordLemmaEmb	Embedding	Embedding of next word's lemma
NextWordPos	Syntactic	POS tag of next word's lemma
PrevWordLemmaEmb	Embedding	Embedding of previous word's lemma
PrevWordPos	Syntactic	POS tag of previous word's lemma
Ne	Semantic	Named Entity tag of word
DepHeadPosition	Position	Position of word within sentence

Table A.2 : The table of Features used in Relation Identification

Feature Nane	Feature Category	Definition
SourceConcept	Lexical	Source concept label
TargetConcept	Lexical	Target concept label
isSameAlignment	AMR-specific	Whether source and target concept are aligned with the same word or not
DepRel	Syntactic	Dependency relation between words aligned with source and target concepts.
SRLRel	Semantic	SRL relation between words aligned with source and target concepts.
sIsType	AMR-specific	Source concept type
tIsType	AMR-specific	Target concept type
sIsMorph	AMR-specific	Whether source concept is morphology derived or not
tIsMorph	AMR-specific	Whether target concept is morphology derived or not
sIsNe	AMR-specific	Whether source concept is entity derived or not
tIsNe	AMR-specific	Whether target concept is entity derived or not
sAlignedWPos	Syntactic	POS tag of word aligned with source concept
tAlignedWPos	Syntactic	POS tag of word aligned with target concept
sAlignedWPersM	Morphologic	Personal marker of word aligned with source concept
tAlignedWPersM	Morphologic	Personal maker of word aligned with target concept
sAlignedWPsessM	Morphologic	Possessive maker of word aligned with source concept
tAlignedWPsessM	Morphologic	Possessive maker of word aligned with target concept
sAlignedWCaseM	Morphologic	Case maker of word aligned with source concept
tAlignedWCaseM	Morphologic	Case maker of word aligned with target concept
sAlignedWVoice	Morphologic	Voice maker of word aligned with source concept
tAlignedWVoice	Morphologic	Voice maker of word aligned with target concept
sAlignedWTense	Morphologic	Tense markers of word aligned with source concept
tAlignedWTense	Morphologic	Tense markers of word aligned with target concept
sAlignedWDeprel	Morphologic	Dependency relation of word aligned with source concept
tAlignedWDeprel	Morphologic	Dependency relation of word aligned with target concept
sAlignedWParentLemma	Lexical	Lemma of head of word aligned with source concept
tAlignedWParentLemma	Lexical	Lemma of head of word aligned with target concept
sAlignedWParentPos	Syntactic	POS tag of word's head aligned with source concept
tAlignedWParentPos	Syntactic	POS tag of word's head aligned with target concept
sAlignedWParetDeprel	Syntactic	Dependency relation of word's head aligned with source concept
tAlignedWParetDeprel	Syntactic	Dependency relation of word's head aligned with target concept
sAlignedWParentPersM	Morphologic	Personal marker of word's head aligned with source concept
tAlignedWParentPersM	Morphologic	Personal marker of word's head aligned with target concept
sAlignedWParentPsessM	Morphologic	Possessive marker of word's head aligned with source concept
tAlignedWParentPsessM	Morphologic	Possessive marker of word's head aligned with target concept
sAlignedWParentCaseM	Morphologic	Case marker of word's head aligned with source concept
tAlignedWParentCaseM	Morphologic	Case marker of word's head aligned with target concept
sAlignedWParentVoice	Morphologic	Voice marker of word's head aligned with source concept
tAlignedWParentVoice	Morphologic	Voice marker of word's head aligned with target concept
sAlignedWParentTense	Morphologic	Tense markers of word's head aligned with source concept
tAlignedWParentTense	Morphologic	Tense markers of word's head aligned with target concept
sAlignedWChildrenDeprel	Syntactic	Set of dependency relations of word's head aligned with source concept
tAlignedWChildrenDeprel	Syntactic	Set of dependency relations of word's head aligned with target concept
sAlignedWChildrenPos	Morphologic	Set of POS tags of word's head aligned with source concept
tAlignedWChildrenPos	Morphologic	Set of POS tags of word's head aligned with target concept
sAlignedWChildrenSRLRel	Morphologic	Set of SRL relations of word's head aligned with source concept
tAlignedWChildrenSRLRel	Morpho-Semantic	Set of SRL relations of word's head aligned with target concept
sNe	Semantic	Named Entity tag of word aligned with source concept
tNe	Semantic	Named Entity tag of word aligned with source concept

Table A.3 : Feature set used in the training of classifiers in the Concept Identification Step

Feature	*TC*	*PC*	*PTC*	*AbsC*	*IC*	*ConjC*	*CauC*	*QC*	*NeC*
SurfaceEmb					✓	✓	✓	✓	✓
LemmaEmb	✓			✓					
Postag	✓	✓	✓	✓	✓	✓	✓	✓	✓
PersMark		✓	✓						
PossessMark		✓	✓						
CaseMark		✓	✓		✓				
Pronoun			✓						
DepHeadLemmaEmb	✓			✓	✓		✓		
DepSibs		✓	✓		✓	✓		✓	
DepChild		✓	✓		✓	✓			
SrlArgSibs		✓	✓		✓				
SrlRel	✓	✓	✓	✓	✓		✓	✓	✓
DepRel	✓	✓	✓	✓	✓	✓	✓	✓	✓
DepHeadPos	✓			✓	✓	✓	✓	✓	
SrlVerbSense		✓							
DepHeadPosessMark					✓				
DepHeadCaseMark								✓	
DepHeadPosition					✓		✓	✓	
VeryProdCI			✓						
sIz/lI					✓				
NextWordLemmaEmb						✓			
PrevWordPos					✓	✓			
Ne	✓			✓	✓				✓

Table A.4 : Feature set used in the training of classifiers in the Relation Identification Step

Feature	*ArgC*	*constCC*	*nonCC*
SourceConcept	✓	✓	✓
TargetConcept	✓	✓	✓
sIsType	✓	✓	✓
tIsType	✓	✓	✓
sIsMorph	✓		✓
tIsMorph	✓		✓
isSameAlignment	✓	✓	✓
DepRel	✓		✓
SRLRel	✓		✓
sAlignedWPos		✓	✓
tAlignedWPos	✓	✓	✓
sAlignedWPersM	✓		✓
tAlignedWPersM	✓		✓
sAlignedWPsessM	✓		✓
tAlignedWPsessM	✓		✓
sAlignedWCaseM	✓		✓
tAlignedWCaseM	✓		✓
sAlignedWVoice	✓		✓
tAlignedWVoice	✓		✓
sAlignedWTense	✓	✓	✓
tAlignedWTense	✓	✓	✓
sAlignedWDeprel			✓
tAlignedWDeprel			✓
sAlignedWParentPos			✓
tAlignedWParentPos			✓
sAlignedWParetDeprel			✓
tAlignedWParetDeprel			✓
sAlignedWChildrenDeprel			✓
tAlignedWChildrenDeprel			✓
sAlignedWChildrenPos			✓
tAlignedWChildrenPos			✓
sAlignedWChildrenSRLRel	✓		✓
tAlignedWChildrenSRLRel	✓		✓
sNe	✓		✓
tNe	✓		✓

APPENDIX B: The Alignment Algorithm

Algorithm 1: Similarity Mapping Algorithm (*simMap*)

Input: $I = List(w_1 \dots w_n)$, $G = (C, R)$, wv, t
Output: M

sim $\leftarrow n \times m$, M $\leftarrow \emptyset$
for $j = 0$ ***to*** n **do**
  $w_j \leftarrow I[j]$
  $v_{wj} \leftarrow wv(wj)$
  for $c_i \in C$ **do**
    $v_{c_i} \leftarrow wv(c_i)$
    if $CosSim(v_{wj}, v_{c_i}) \geq t$ **then**
      sim$[w_j][c_i] \leftarrow CosSim(v_{wj}, v_{c_i})$
    else if $mFuzzy(v_{wj}, v_{c_i} \geq 0.95$ **then**
      sim$[w_j][c_i] \leftarrow mFuzzy(v_{wj}, v_{c_i})$
    end
  end
end
for $j = 0$ *to* n **do**
  $concepts \leftarrow mapping(sim, I, j, G)$
  for each c_i *in* $concepts$ **do**
    M[j] $\cup \{< I[j], c_i >\}$
  end
end
Function *mapping (sim,I,j,G)*:
  $w_j \leftarrow I$[j]
  $s \leftarrow max$(sim$[w_j]$)
  for $c_l \in C$ **do**
    $w_k \leftarrow argmax$(sim$[:, c_l]$)
    if $w_k = w_j$ **then**
      $cands \cup \{c_l\}$ if $s =$sim$[w_k, c_l]$
    end
  end
  if $length(cands) \geq 2$ **then**
    $cands \leftarrow disambugate(cands, I$,j,$G)$
  end
  return *cands*
end

```
34 Function disambugate (candidates,I,i,G):
35     max ← 0, v ← ∅
36     for each c in candidates do
37         if c is verb frame then
38             return candidates
39         else
40             children ← getChildren(c,G)
41             prev ← {I[i−1]} , next ← {I[i+1]}
42             parents ← getParent(c,G)
43             overlap ← { prev ∩ children } ∪ { next ∩ parents }
44             if length(overlap)≥ max then
45                 v ← candidate , max ← length(overlap)
46             end
47         end
48     end
49     return { v }
50 end
```

Algorithm 2: Alignment Algorithm

Input: $I = List(w_1 \dots w_n)$, $G = (C,R)$
Output: $Alignments$
$M \leftarrow simMap(I,G)$
$M \leftarrow sort(M)$
$T \leftarrow removeReent(G)$
$Alignments \leftarrow \emptyset$
for $j = 0$ ***to*** n **do**
    $pl_j \leftarrow M[j]$
    for $< w_j, c_i > \in pl_j$ **do**
        $Alignments[j] \cup \{c_i\}$
        $Alignments[j] \cup getVisited(c_i,T,M)$
    end
end
$Alignments \leftarrow postprocess(Alignments)$
Function *getVisited* (c,T,M)**:**
    $visited \leftarrow \emptyset$
    $n \leftarrow NeighInAllowedPath(c,T,M)$
    for $n_i, \in n$ **do**
        if $< \forall w, n_i > \notin M$ **then**
            $visited \cup getVisited(n_i,T,M)$
    end
    return $visited$
end

Table B.1 : The table of Concept-Allowed Paths pairs used in the Alignment Algorithm

Concepts	Allowed Paths
person	:ARG0, :ARG0-of, :name, :mod, :age, :source, :location
thing	:ARG1, :ARG1-of, :name
city	:name
name-entity concepts	:name
be-located-at-91	:ARG2, :ARG2-of
be-temporally-at-91	:ARG2, :ARG2-of
have-rel-role-91	:ARG2, :ARG2-of
have-org-role-91	:ARG2, :ARG2-of
have-degree-91	:ARG3, :ARG3-of
have-quant-91	:ARG3, :ARG3-of
monetary-quantity	:quant, :value
rate-entity	:quant, :value
ordinal-entity	:quant, :value
temporal-quantity	:unit
include-91	:ARG2, :ARG2-of
date-entity	:day, :year, :month, :dayperiod, :weekday, :century
have-li-91	:ARG2, :ARG2-of
rate-entity-91	:ARG2, :ARG2-of
instead-of-91	:ARG2, :ARG2-of
date-interval	:op1, :op2
have-purpose-91	:ARG2, :ARG2-of

CURRICULUM VITAE

Name SURNAME: Kadriye Elif ORAL

EDUCATION:

- **B.Sc.:** 2016, Marmara University, Engineering Faculty, Electrical Enginering and Electronics

PROFESSIONAL EXPERIENCE:

- May 2022 - continue, NLP Lead Scientist at Hepsiburada
- March 2020 - May 2020, Senior Data Scientist at Hepsiburada
- June 2019 - March 2020, Data Scientist at Doğuş Technology
- July 2017 - June 2019, Assistant Research Engineer at Huawei Research Center

PUBLICATIONS, PRESENTATIONS AND PATENTS ON THE THESIS:

- **Oral, Elif**, Ali Acar, and Gülşen Eryiğit. "Abstract meaning representation of Turkish." Natural Language Engineering (2022): 1-30. (Article Instance)
- **K. Elif Oral** and Gülşen Eryiğit. 2022. AMR Alignment for Morphologically-rich and Pro-drop Languages. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop, pages 143–152, Dublin, Ireland. Association for Computational Linguistics. (Presentation Instance)
- **E. Oral**, Gülşen Eryiğit "The Impact of Pre-trained Language Models on Turkish Semantic Role Labelling," 2022 30th Signal Processing and Communications Applications Conference (SIU), 2022, (Presentation Instance)