Transitions all the Way Down: From Characters to Full Document Annotation in One System

Yuval Pinter
Department of Compuer Science
Ben-Gurion University of the Negev
Beer Sheva, Israel

Miryam de Lhoneux
Department of Computer Science
KU Leuven
Belgium

Working Groups 1 & 3

1 Introduction

Existing Natural Language Processing (NLP) systems that process raw text into complex linguistic structures are built on what is known as the "NLP pipeline". A pre-processing step produces sentences divided into tokens, then a Part-of-Speech tagger is applied to this clean token representation, followed by (a lemmatizer and) a syntactic parser, followed by a semantic parser, followed by supersentential analysis such as a discourse parser, a summarizer, or a coreference system (e.g., Straka et al., 2016). While systems exist that fuse several of these steps together to form, e.g. a joint tagger-parser (Bohnet and Nivre, 2012), to the best of our knowledge none has sought a single framework that starts at the character level and annotates an entire document in one pass.[1]

We propose a single transition-based model to jointly perform multiple levels of the NLP pipeline: tokenization, word segmentation, sentence detection, part-of-speech tagging, lemmatization, morphological tagging, syntactic parsing, and discourse parsing. Our schema negates the need for trained word-level embeddings or shared tokenization modules, while also leveraging annotated resources at the morphological and syntactic levels. We have begun implementing a parser and conducting experiments on several languages where required resources are available including Hungarian, English, Turkish, and Swedish; our results are still preliminary.

2 Framework Sketch

Our model starts at the character level and provides all levels of annotations for a typical NLP pipeline, or more specifically, all levels of annotations contained in Universal Dependencies (UD) treebanks (Nivre et al., 2020). It follows a transition-based shift-reduce mechanism, where characters from the buffer are incrementally shifted onto the stack, and occasionally merged through reduce operations into objects belonging to higher representational layers to ultimately produce annotations in the output structure. Each representational layer is subject to its own actions when corresponding members are exposed on the stack. For example, when the top stack items are *words*, syntactic tree operations (like in a classic shift-reduce parser) are licensed for the next step.

2.1 Components

The most basic component of our model is the **character buffer**, not necessarily limited to a single sentence (since our model supports supersentential annotation). We require this buffer to support lookahead of some (possibly constant) number of characters ahead of our current location, the exact amount of which can be determined by the implementation details, and may also be affected by incorporation of pre-processing such as a bidirectional RNN pass over the data.

The other essential component of the parser is a **stack**, whose elements may instantiate a mix of different types as the parsing operation progresses: character elements are joined into morphemes, which in turn join into words, which then form the syntactic structure of the output.

The **output annotations** provided by the transition actions include not only dependency arcs but also morphosyntactic descriptions (MSD) and POS labels, corresponding to individual words.

[1] Of the tasks omitted in our effort, most notable is NER. We intend to pursue it in future work.

#	form	lemma	pos	head	relation	morphosyntactic attributes
1-3	BBTYM					
1	B		PREP	3	case	
2	H		DT	3	det	
3	BTYM	BYT	NOUN	0	root	Number=Plural;Def=Definite
4-5	HGDWLYM					
4	H		DT	5	det	
5	GDWLYM	GDWL	ADJ	3	amod	Number=Plural;Def=Definite

Table 1: Desired tagged output for the Hebrew fragment 'BBTYM HGDWLYM'.

2.2 Transitions

The basic operation, SHIFT, moves a character from the buffer to the stack. In case of a non-syntactic character encountered (typically a space, but not all spaces), DISCARD is used to denote an ignored character. Three distinct REDUCE operations form morphemes, words, and tokens, and place them on the stack, replacing their constituents. Two of the REDUCE operations, namely MORPH and WORD, produce an annotation for the final output, corresponding to the layer of the reduced object—an MSD or a POS tag, respectively. The role of REDUCE-TOKEN is to account for multi-word tokens, and their creation does not entail tree-level annotations. Two transitions, ARC-LEFT and ARC-RIGHT, are responsible for the production of the syntactic dependency tree. RED-SENT is a super-sentential REDUCE operation responsible for discourse-level marking, if such exists in the data. It allows for annotation of a discourse marker signifying the relation between the sentence and the one following it.

3 Walkthrough

The Hebrew sequence 'BBTYM HGDWLYM' '*in the large houses*'[2] contains multiple syntactic decisions that can help explain the workings of the suggested model. First, it exhibits **syntactic fusion** where the case marker 'B' '*in*' and the definite marker 'H' '*the*' are attached to the adjacent content word. Moreover, the 'B' implicitly contains a definite marker, not directly observable in text but inferred by the rules of Hebrew definite agreement and expressed when spoken. Finally, the morpheme 'YM' '*-pl*' present in both tokens is a "conventional" plural marker. The desired output is depicted in reduced CoNLL-U form in Table 1.

Scanning from left to right, the sequence of actions taken by an oracle character-level parser would be the following:

SHIFT, RED-MORPH$_{\emptyset}$, RED-WORD$_{\text{PREP}}$, RED-MORPH$_{\textit{Def=Definite}}$, RED-WORD$_{\text{DT}}$, SHIFT, SHIFT, RED-MORPH$_{\textit{Lemma='BYT'}}$, SHIFT, SHIFT, RED-MORPH$_{\textit{Number=Plural}}$, RED-WORD$_{\text{NOUN}}$, ARC-LEFT$_{\text{DET}}$, ARC-LEFT$_{\text{CASE}}$, RED-TOK, DISCARD, SHIFT, RED-MORPH$_{\textit{Def=Definite}}$, RED-WORD$_{\text{DT}}$, SHIFT, SHIFT, SHIFT, SHIFT, RED-MORPH$_{\textit{Lemma='GDWL'}}$, SHIFT, SHIFT, RED-MORPH$_{\textit{Number=Plural}}$, RED-WORD$_{\text{ADJ}}$, ARC-LEFT$_{\text{DET}}$, ARC-RIGHT$_{\text{AMOD}}$, ARC-RIGHT$_{\text{ROOT}}$, RED-TOK, RED-SENT$_{\emptyset}$

As morphemes are reduced, their MSDs are accumulated and assigned to the content word produced at the next RED-WORD. The lemmas are produced by stem morphemes, with an open-vocabulary value scheme for the *Lemma* attribute. Note that these might not correspond to the exact stem (see BT/BYT).

4 Model Implementation

Our parser implementation contains multiple trainable distributed representations, for: (a) all types of linguistic constituents (characters, morphemes, words, tokens, sentences); (b) annotation classes (POS tags, MSDs, dependency labels, discourse types); and (c) the current states of the various data structures (character buffer, lookahead buffer, stack, trees).

We implement feedforward neural **combiners** for the different representation layers, which operate once a REDUCE or ARC transition calls them.

The main challenge in creating an **oracle** for our framework is aligning subword morphemes with MSD annotations, for which we create rules based on the MorphyNet resource (Batsuren et al., 2021), available for about a dozen languages.

[2]Hebrew transliterated for convenience, character-to-character.

References

Khuyagbaatar Batsuren, Gábor Bella, and Fausto Giunchiglia. 2021. MorphyNet: a large multilingual database of derivational and inflectional morphology. In *Proceedings of the 18th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 39–48, Online. Association for Computational Linguistics.

Bernd Bohnet and Joakim Nivre. 2012. A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1455–1465, Jeju Island, Korea. Association for Computational Linguistics.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. Universal Dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4034–4043, Marseille, France. European Language Resources Association.

Milan Straka, Jan Hajič, and Jana Straková. 2016. UDPipe: Trainable pipeline for processing CoNLL-U files performing tokenization, morphological analysis, POS tagging and parsing. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4290–4297, Portorož, Slovenia. European Language Resources Association (ELRA).