Part-of-Speech Tagging using Contextual Word Embeddings

Juhani Dickinson
jdickin9@uwo.ca

Paul Moore
email address or ORCID

***Abstract*—ABSTRACT**

***Index Terms*—pos tagging, part of speech, nlp, word embedding, contextual word embedding**

I. Introduction

A. *Parts of Speech*

In a sentence, each word has a syntactic role to play. In the sentence “The brown dog runs quickly towards food.”, “dog” denotes a thing, “brown” modifies or describes “dog”, “runs” denotes an action, and so on. These syntactic roles are commonly referred to as “parts of speech”, and are mapped to words using part-of-speech tags. However, this is not a one-to-one mapping. Many words have multiple meanings, and in many cases, these different meanings play different syntactic roles, mapping them to different parts of speech. For example, the word “orange”, spelled and pronounced identically, can be either a noun or an adjective, as seen in sentences (1) and (2).

“I often eat an <u>orange</u> after working out.” (Noun) (1)

“The <u>orange</u> car has arrived.” (Adjective) (2)

Figuring out what part of speech a word has is not an unsolvable task, however. Where a word is in the sentence and what words surround it play a major role in disambiguating between syntactic roles. In (1), the word after “orange” is “after”, which is an adverb. “Orange” is either an adjective or a noun, and since the following word is an adverb, it cannot be an adjective. In (2), the words surrounding “orange” are “the” and “car”, which are a determiner and a noun respectively. In this position, “orange” describes “car”, making it an adjective.

B. *Part-of-Speech (POS) Tagging*

The task of POS tagging (often simply called tagging) is as follows: given a sentence, label each word in the sentence with the POS tag that describes its syntactic role, seen below for sentence (3).

(3)										
	Can	you	<u>spot</u>	the	large	<u>spot</u>	on	your	nose	?
	AUX	D	VB	D	ADJ	N	PP	D	N	.

What makes the task of tagging challenging is the contextual requirement: without it, both instances of “spot” in sentence (3) would be tagged as either VB or N. Because of this, non-naive tagger implementations leverage some level of context to perform their task. Simple implementations like n-gram taggers assign tags to words based on both the word itself and the tags of the preceding n words.

C. *Word Embeddings*

A common issue with simple tagging implementations is resistance to parallelisation. Any operation below the level of sentence must be linear, as the tag of the current word depends on the tag(s) of the previous words. To get around the linear-time requirement, some taggers use word embeddings, which represent words as vectors. This embedding function takes into account the surrounding words, so that the meaning and context of each word is stored in the embedding. Word embeddings are classifier models usually trained using one of two methods: predict target word from context (Continuous Bag of Words, CBOW) and predict context from target word (Skipgram). In CBOW, the words surrounding the target word are mapped to one-hot vectors, which are each then projected down to a dense vector by the classifier model and averaged together. This averaged vector is then used to predict the target word. In Skipgram, the target word is mapped to a one-hot vector and then projected down to a dense vector by the classifier. This is done for multiple target words, the dense vectors of which are averaged together and used to predict the context in which the target words appear. Using embeddings, tagging can be parallelised at the level of individual words.

1) Static Word Embeddings: While word embeddings take context into account by design, static word embeddings take the naive approach to context. These mapping functions collect all contexts for a given word and uses them to generate a single embedding per word. Often, this approach fails to take into account that words can have different meanings (“river <u>bank</u>” vs. “<u>bank</u> account”) or different syntactic roles. This means that a singular word embedding must hold information on every possible context for any given word, which causes problems when the different meanings of a word are not syntactically or semantically close to one another.

2) Contextual Word Embeddings: Contextual word embeddings solve the syntactic and semantic problems that static word embeddings have by moving from word embeddings to word token embeddings. Now, words with multiple meanings or syntactic roles are split into multiple word tokens, one for each use. Contextual embeddings map these tokens to vectors, preserving specific meaning. With contextual embeddings, a word like “spot” is first split into two tokens, such as “spot–noun” and “spot–verb”, and these tokens are embedded separately. While contextual embedding systems are more computationally demanding to train and use, their contextaware nature often leads to increased performance in context-sensitive tasks such as POS tagging.

II. Related Work

pass

III. Motivation

Part-of-speech tags are useful and sometimes key pieces of information in many natural language processing (NLP) tasks such as text-to-speech (TTS), word-sense disambiguation, marking word order, and named entity recognition (¡ref¿). In TTS applications, the pronunciation of a word can change depending on what its syntactic role is or sometimes even what tense it is. The word "resume" is pronounced differently based on whether it is meant as "continue doing" or as "a document detailing professional experience", which can be easily disambiguated based on whether it acts as a verb or a noun. This disambiguation-by-tag is also used in word-sense disambiguation, like in sentence (3). In a search-engine setting, understanding whether "spot" refers to an activity or a visual marker allows for more accurate search results.

A. Benefits for NLP in Low-Resource Languages

While high-resource languages like English no longer use POS tagging in some popular NLP applications like AI assistants by making use of the massive collections of curated data now available, most languages do not have the luxury of large, high-quality datasets. In these low-resource languages, POS tags are still critical for NLP tasks. For a task like machine translation, (¡ref¿) found that using linguistic features derived from POS tags improved the translation performance not only between the low-resource language pair of Thai and Myanmar but also between Thai and English and Myanmar and English. While the highest accuracy was found on English-Thai translations, likely due to both languages having a Subject-Verb-Object sentence structure, the English-Myanmar and Thai-Myanmar pairs both saw improvements over the baseline.

While some low-resource languages have POS taggers or have the resources available for one to be reasonably made, many such languages do not. In such cases, POS taggers for high-resource languages are still quite useful. In low-resource languages where training data is scarce, one established method of finding training data is to use cross-linguistic transfer to automatically generate POS training data. This is done by lining up texts that appear in both a high-resource language and the target low-resource language and running POS taggers and parsers for the high-resource language (¡ref¿). The results from these taggers and parsers are then projected onto the low-resource text. With an advanced projection method such as Graph Label Propagation (GLP) proposed by (¡ref¿) to transfer tags from only English, tagging accuracy on a variety of languages averaged 80.6%. Notably, methods such as GLP can leverage multiple parallel texts from high-resource languages to increase performance even further, with tests on the same variety of languages achieving an accuracy of 84.0%. However, leveraging multiple languages simultaneously is not always possible for certain low-resource languages due to the lack of available parallel data.

As well as being useful, the efficacy of cross-linguistic transfer is predictable. Reference (¡ref¿) found that language families and morphological structures have a major impact on the performance of cross-linguistic performance. Specifically, performance was much higher when the source model for cross-linguistic transfer was in the same language family as the target language. This was especially true of morphologically-rich languages. Even outside of the same family, cross-linguistic transfer had better performance when moving between morphologically-rich languages, contrasting with morphologically-poor languages experiencing steeper drops in performance in similar cross-family settings. Importantly, no data from the target language was used for training or fine-tuning in these tests, which indicates that a target language being low-resource does not impact the efficacy of cross-linguistic transfer.

pass

B. Benefits of an English POS tagger

Finding a good POS tagger for English, a high-resource language, has clear benefits for linguistically-related low-resource languages due to the success of cross-linguistic transfer. One example is Scots, a language in the Anglic family currently considered "vulnerable" by the UNESCO Atlas of the World's Languages in Danger (¡ref¿). Research suggests that NLP can support and help revitalise endangered languages (¡ref¿) through machine translation, TTS, and integration into learning materials. With competent machine translation, more materials can be translated into Scots, which encourages learning and expands which facets of life can be interacted with in Scots. TTS has applications when paired with translation and learning tools, and brings one facet of accessibility into the language as well. Finally, machine-based language learning apps like Duolingo have made use of NLP and AI to provide more language learning content (¡ref¿), and members of the Scots speaker community can make use of POS tagging directly to help with learning Scots linguistics.

To this end, we will test multiple prominent word embedding models on the same English dataset in order to determine the best model. Knowing the best model ¡–FINISH THIS–¿

IV. Methods

pass

V. Experimental Results

pass

VI. Conclusion

pass

References

[1] Pass