Compositional Morpheme Embeddings with Affixes as Functions and Stems as Arguments

Daniel Edmiston
University of Chicago
danedmiston@uchicago.edu

Karl Stratos
Toyota Technical Institute at Chicago
stratos@ttic.edu

Abstract

This work introduces a linguistically motivated architecture, which we label StAffNet, for composing morphemes to derive word embeddings. The principal novelty in the work is to treat stems as vectors and affixes as functions over vectors. In this way, our model's architecture more closely resembles the compositionality of morphemes in natural language. Such a model stands in opposition to models which treat morphemes uniformly, making no distinction between stem and affix. We run this new architecture on a dependency parsing task in Korean—a language rich in derivational morphology—and compare it against a lexical baseline, along with other sub-word architectures. StAffNet shows competitive performance with the state-of-the-art on this task.

1 Introduction

This work proposes a novel architecture for the composition of morphemes to derive word embeddings. The architecture is motivated by linguistic considerations and is designed to mirror the composition of morphemes in natural language. This means making a hard distinction between affix and stem (e.g. between content morphemes like stem *dog* and functional morphemes like plural affix *-s* in the word *dogs*), and recognizing the function-argument relation between them. We reflect this in our architecture by treating stems as vectors in $\mathbb{R}^n$, and affixes as functions (either linear or non-linear, depending on model) from $\mathbb{R}^n$ to $\mathbb{R}^n$. Given the importance of stems and affixes in the architecture, we label it St(em)Aff(ix)Net.

We test the viability of the linguistically motivated StAffNet on a dependency parsing task in Korean—a language rich in derivational morphology. Here, we achieve promising results for infusing explicit linguistic analyses into NLP architectures. Specifically, the architecture achieves results which significantly outperform simple word-embedding baselines, and are competitive with other sub-word architectures which constitute the current state-of-the-art for this task in Korean (Stratos, 2017).

We therefore submit the following as our contributions:

- We introduce a novel architecture for the composition of word-embeddings which is explicitly designed to mirror composition of morphologically complex words in natural language.
- Our novel architecture achieves state-of-the-art performance in every case (see Table 1), suggesting linguistic structure can be viable for real-world NLP tasks.

2 Related Work

This work falls under a large body of work on incorporating linguistically sound structures into neural networks for more effective text representation. One such line of work is sub-lexical models. In these models, word representations are enriched by explicitly modeling characters (Ma and Hovy, 2016; Kim et al., 2016) or morphemes (Luong et al., 2013; Botha and Blunsom, 2014; Cotterell et al., 2016). For languages with complex orthography, sub-*character* models have also been proposed. Previous works consider modeling graphical components of Chinese characters called radicals (Sun et al., 2014; Yin et al., 2016) and syllable-blocks of Korean characters—either

Proceedings of the Workshop on the Relevance of Linguistic Structure in Neural Architectures for NLP, pages 1–5
Melbourne, Australia, July 19, 2018.

as atomic (Choi et al., 2017) or as non-linear functions of underlying *jamo* letters through Unicode decomposition (Stratos, 2017).

The present work also aims to incorporate sub-word information into word embeddings, and does so by modeling morphology. However, this work differs from those above in the means of composition, as our method is based principally on function application. Here, we take derivational morphemes (i.e. affixes) as functions, and stems as arguments. Broadly speaking, this work can be seen as an extension of Baroni et al. (2014)'s compositional distributional semantic framework to the sub-word level. At a more narrow level, our work is reminiscent of Baroni and Zamparelli (2010), who model adjectives as matrices and nouns as vectors, and work like Hartung et al. (2017), which seeks to learn composition functions in addition to vector representations.

3 Architecture and Linguistic Motivation

The intuition behind the decision to treat stems and affixes differently is that to do otherwise is to miss a key linguistic generalization with regard to the composition of complex words. Furthermore, we argue that to include stems and affixes in the same space for comparison is akin to doing the same for, say, real numbers and functions over real numbers. In the same way that the squaring operation is defined as a function of its input, we argue that an affix has meaning only insofar as the effect it produces on its stem.

Regarding the behavior of morpheme composition in natural language, we know that stems can compose to form compounds, and affixes can attach successively to a stem. However, affixes cannot exist in isolation—they must attach to a stem. We seek for our architecture to display each of these properties: compounding, successive affix attachment, and inability to represent an affix on its own. Therefore, in order to induce compositional morpheme representations, we learn not only vectors for stems, but also a weight matrix and bias for each affix.

To accomplish this, we use the *Komoran* part-of-speech tagger included in the *KoNLPy*[1] toolkit, and have a trained theoretical linguist separate out the stem parts of speech from the affix parts of speech. We then parse Korean words into constituent stems and affixes, and compute the compositional representation of a word from these constituent parts using a dynamic neural-network architecture. The architecture can be conceptually broken into three steps: (i) decomposing the word into its constituent stems and affixes, (ii) computing the composite stem representation, and then (iii) iteratively applying (equivalent to function composition) the affix functions over the stem representation.

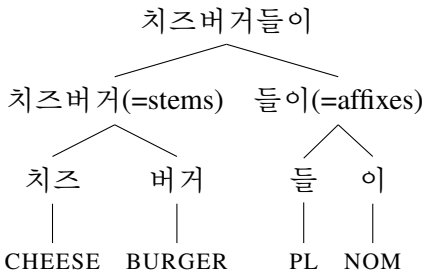

Figure 1: Decomposition of 치 즈 버 거 들 이 (=*cheese.burger*-PL.NOM)

To illustrate how the architecture works in detail, we consider the morphologically complex Korean word for "*cheeseburgers*" marked with nominative case: 치즈버거들이(=*cheese.burger*-PL.NOM).

First, the word is decomposed by our part-of-speech tagger into a list of stems, [*cheese*, *burger*], and a list of affixes, [PL, NOM]. This decomposition is as in Figure 1. Given a list of stems, we decide how to construct a stem representation made from the elements in that list. If the stem list has only a single member, we simply return that stem's representation as the full stem representation.

Since *cheese.burger* is a compound stem, we must go through the step of constructing a composite stem representation. To do this, we first run a vanilla bi-directional RNN over the stem sequence (the choice of a vanilla BiRNN rather than a more powerful mechanism capable of capturing long distance dependencies rests on the apparent fact that Korean morphological dependencies are strictly local, lacking phenomena like circumfixes or non-concatenative morphology). This produces an intermediate output for each stem in the sequence, $e^{<t>}$, which we weight and then sum together for the composite stem representation.

In order to calculate the proper weighting for each stem, $w^{<t>}$, we compare each output of the RNN via cosine-similarity with a pre-trained embedding of the full sequence of

[1] Documentation for which is available at `konlpy.org`.

stems, in this case 치즈버거, or *cheeseburger* with no affixes attached.[2] This gives us scores $s^{<1>} = cos(cheeseburger, cheese)$ and $s^{<2>} = cos(cheeseburger, burger)$. We softmax the sequence $[s^{<1>}, s^{<2>}]$, giving us our weights $w^{<1>}$ and $w^{<2>}$. The composite stem representation is then the sum of our weighted intermediate scores, i.e. $\sum_t w^{<t>} \cdot e^{<t>}$.

Presumably, since the word *cheeseburger* acts more like *burger* than *cheese*, *burger* will receive a higher cosine similarity and thus be weighted more. In this way, our system has a natural and dynamic way of weighting stems.

Now that we have a composite stem representation, we can feed it iteratively as an argument to the affix list. Here, each affix is represented either as a non-linear function $\lambda x.tanh(W \cdot x + b)$ in the model we call StaffNet Non-Linear, or as a linear function $\lambda x.W \cdot x + b$ in the model we call StaffNet Linear (though there is still non-linearity in the RNN calculating the composite stem representations). The models are otherwise identical, and in each case W and b are learnable parameters. The StaffNet Non-Linear computation graph for the example 치즈버거들이(= *cheese.burger*-PL.NOM), or *cheeseburgers* in the nominative case, is as in Figure 2.

4 Performance on Parsing Task

In order to test the efficacy of our composition method, we ran experiments for both our linear and non-linear models on a dependency parsing task using a publicly available Korean treebank (McDonald et al., 2013).[3] Word vectors were composed as described above in 100 dimensions, and then these representations were inserted into the BiLSTM model of Kiperwasser and Goldberg (2016).[4] We then compared our results to the original results in McDonald et al. (2013) and to those reported in Stratos (2017) for various sub-word architectures also run with Kiperwasser & Goldberg's parser. These results were all trained on a training set of 5,425 sentences over 30 epochs, with the best model being chosen from a dev set of 603 sentences. Finally, the test set consisted of 298 examples. The results are summarized in Table 1.

[2]The pre-trained embeddings are *word2vec* (Mikolov et al., 2013), skip-gram-induced embeddings with a window of 5.

[3]https://github.com/ryanmcd/uni-dep-tb

[4]https://github.com/elikip/bist-parser

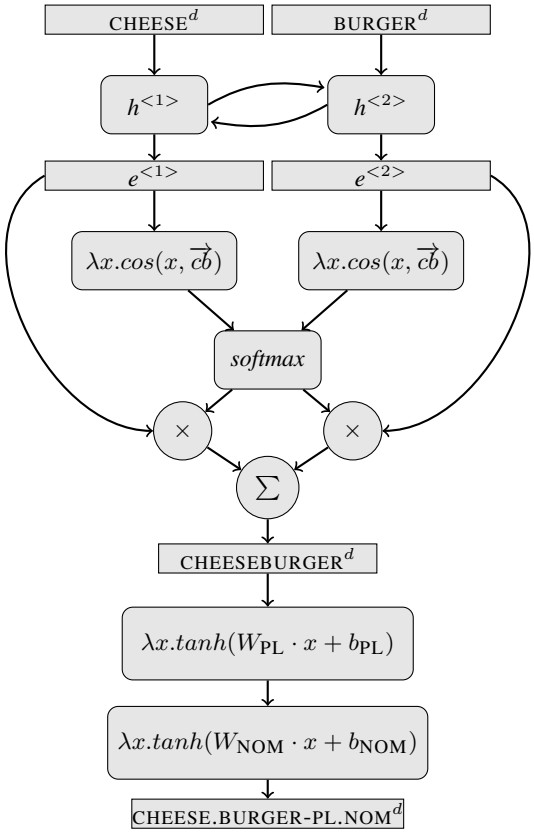

Figure 2: Composition of complex word *cheese.burger*-PL.NOM

System	embedding	UAS	LAS
McDonald13	word	71.22	55.85
K&G16	word	90.00	82.77
Stratos17	syllable	94.75	90.81
	letter	94.59	90.77
	syllable/letter	94.79	91.19
	word/syl/let	95.17	92.31
StaffNet Non-Linear	stem & affix	95.17	92.89
	word/stem & affix	95.06	92.93
StaffNet Linear	stem & affix	**95.48**	**93.43**
	word/stem & affix	95.17	93.32

Table 1: System comparison

A quick examination of Table 1 shows that our systems significantly outperform lexical baselines, showing that the incorporation of sub-word information in a linguistically motivated fashion can demonstrate good performance on NLP tasks. Furthermore, our models are highly competitive with competing sub-word architectures. Both StaffNet models achieve virtually identical results with those in Stratos (2017), with the StaffNet Linear model slightly edging out the others. It is perhaps surprising that neither the

addition of non-linearity to affix transformations, nor the concatenation of lexical representations to STAFFNET representations appear to make any significant change in results.

It is worth noting that the jump provided by incorporating sub-word information is significantly higher for LAS than UAS when compared to the lexical baselines.[5] This could be due to the simple fact that there is more room for improvement in LAS than UAS, but we speculate below on a potentially more interesting explanation based on the apparent role of (certain types of) morphology in natural language.

5 Discussion

It is no surprise that incorporating sub-word information outperforms more basic, lexically tokenized systems, and given the results in Table 1, it is easy to be optimistic with regard to the idea of incorporating sub-word information in a linguistically motivated fashion.

But what's interesting is not so much the fact that STAFFNET outperformed lexical baselines, it is how it did it. In the best case, STAFFNET outperformed K&G's BiLSTM model with simple word embeddings by 5.48 in UAS, but by 10.66 in LAS. We hypothesize that this jump was not simply due to there being more room for improvement in LAS. Rather, we speculate that this significant improvement in LAS was due to the apparent role of certain types of morphology in natural language, particularly case morphology. The role of case morphology in natural language is to mark relations between syntactic constituents. An explicit marker for syntactic relations like case morphology is likely to aid in a task like LAS, where the goal is to label these syntactic relationships. This is especially true for Korean, where case morphology is both regular and frequent. We hypothesize that morphologically aware architectures like STAFFNET are well suited to leverage this information when labelling arcs.

It may be asked why the syllable-based embeddings of Stratos (2017) also showed such a strong improvement over lexical baselines in LAS versus UAS (82.77 to 90.81 and 90.00 to 94.75), but this may have to do with the nature of the lexical makeup of the Korean language. The vocabulary of the Korean language is, depending on dictionary, made up of between 52% and 69% of words of Chinese origin, known as *hanja*. These *hanja* are single-syllable, meaning-bearing units, meaning it's very likely that syllable embeddings implicitly capture a great deal of meaningful lexical content in a way that similar sub-word architectures (e.g. *fastText*; Bojanowski et al., 2016) in languages like English cannot. Furthermore, many of the most common case-markings in Korean consist of a lone syllable, meaning this system too would have a strong advantage at implicitly capturing case meaning, and therefore have an advantage when labelling arcs. It is less clear what to make of the effectiveness of compositional letter embeddings for Korean, though this representation has by far the smallest number of parameters and yet still shows state-of-the-art performance, making it the most practical choice of sub-word architecture for Korean.

6 Conclusion and Future Work

This paper introduced a novel architecture, STAFFNET, for composing word embeddings using morphemes as atomic units. It was novel in that it made a distinction between stem representations and affix representations, the former being vectors and the latter being (non-)linear functions over those vectors. The intuition is to more closely mimic how natural language is thought to handle morphological composition and make a distinction between the lexically contentful and the functional. We tested the mettle of this architecture in a dependency parsing task, where it showed very strong results, slightly outperforming the state-of-the-art.

In addition to the practical import of achieving state-of-the-art performance in a novel way, we argue that this exercise has been both useful and enlightening from a linguistic viewpoint. Useful in that a linguistically motivated system shows strong performance and emerges as a candidate for sub-word architectures (at least in morphologically rich languages like Korean), and enlightening in that the manner in which these compositional morphemes improve upon the lexical baseline is disproportionate in helping the parser label its arcs. We speculate that this is because the nature of relations between syntactic entities is often reflected in the morphology, and this is especially

[5] Labeled Attachment Score, or LAS, refers to the percentage of correct assignments of words to their heads along with the correct label. Unlabeled Attachment Score, or UAS, refers only to the percentage of correct attachments, regardless of label.

true in languages rich in case morphology.

We see future work going forward in any of three directions:

- Improving upon the system described here; we rely on the *Komoran* part-of-speech tagger for decomposing words—is there a better way to do this? Was the choice of vanilla BiRNN for composite stem representation a good one? Could we achieve even higher results with more sophisticated networks?
- Testing this architecture on other languages. Korean is rich in case morphology. Would our system show such improvement over lexical baselines on languages with more impoverished morphology?
- Can this type of architecture be successful at the level of syntax, as a means of deriving compositional sentence embeddings?

Acknowledgments

The authors would like to thank John Goldsmith and Liwen Zhang for stimulating discussion related to STAFFNET.

References

Marco Baroni, Raffaella Bernardi, and Roberto Zamparelli. 2014. Frege in space: A program for compositional distributional semantics. Linguistic Issues in Language Technology, 9:241–346.

Marco Baroni and Roberto Zamparelli. 2010. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, pages 1183–1193.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information.

Jan A Botha and Phil Blunsom. 2014. Compositional morphology for word representations and language modelling. In ICML, pages 1899–1907.

Sanghyuk Choi, Taeuk Kim, Jinseok Seol, and Sanggoo Lee. 2017. A syllable-based technique for word embeddings of Korean words. In Proceedings of the First Workshop on Subword and Character Level Models in NLP, pages 36–40.

Ryan Cotterell, Hinrich Schütze, and Jason Eisner. 2016. Morphological smoothing and extrapolation of word embeddings. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, volume 1, pages 1651–1660.

Matthias Hartung, Fabian Kaupmann, Soufian Jebbara, and Philipp Cimiano. 2017. Learning compositionality functions on word embeddings for modelling attribute meaning in adjective-noun phrases. In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, volume 1, pages 54–64.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016. Character-aware neural language models. In Thirtieth AAAI Conference on Artificial Intelligence.

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional lstm feature representations. In Transactions of the Association for Computational Linguistics, volume 4, pages 313–327.

Thang Luong, Richard Socher, and Christopher D Manning. 2013. Better word representations with recursive neural networks for morphology. In CoNLL, pages 104–113.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.

Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, Claudia Bedini, Núria Bertomeu Castelló, and Jungmee Lee. 2013. Universal dependency annotation for multilingual parsing. In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, volume 2, pages 92–97.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems, pages 3111–3119.

Karl Stratos. 2017. A sub-character architecture for Korean language processing. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 721–726, Copenhagen, Denmark. Association for Computational Linguistics.

Yaming Sun, Lei Lin, Nan Yang, Zhenzhou Ji, and Xiaolong Wang. 2014. Radical-enhanced Chinese character embedding. In International Conference on Neural Information Processing, pages 279–286. Springer.

Rongchao Yin, Quan Wang, Rui Li, Peng Li, and Bin Wang. 2016. Multi-granularity Chinese word embedding. In Proceedings of the Empiricial Methods in Natural Language Processing.