Unsupervised Language Learning Final Assignment

Akash Raj Komarlu Narendra Gupta
11617586

Jack Harding
11623608

1 Introduction

In this paper, we briefly outline the conceptual backdrop to word embeddings. We describe two models, Skipgram and Embed-Align, and their extrinsic evaluation using the SentEval toolkit. A key contribution of our paper is comparing the effects of distinct methods of composing sentence embeddings (from word embeddings) on each of the models. We find that different models benefit from different composition methods, and discuss possible reasons for this.

2 Background

Word embeddings represent natural language words as real-valued vectors in a high-dimensional space. Most extant word embedding models rely on the *Distributional Hypothesis* (DH), which claims that the meaning of a word correlates with its distributional properties (such as the words which surround it) in human-generated corpora (Harris, 1954). In *count-based models*, individual dimensions of word embeddings correspond to occurrences of a (generally hand-crafted) semantic feature, such as co-occurrence with another word in the vocabulary or a syntactic dependency relation (Miller and Charles, 1991). In *prediction-based models*, words are represented by an approximation to a function which predicts the contexts of the word. Typically, a neural network is used as the function approximator; word embeddings are then weights of the trained network (Mikolov et al., 2013). Although the resultant vectors are dense (making them more amenable to manipulation than sparse count-based embeddings), the semantic features learned during the prediction task are latent, making prediction-based embeddings difficult to interpret.

There are two major problems with contemporary word embeddings. Firstly, they are unable to capture polysemy, since each word is represented by a single vector. Secondly, they inherit a problem with DH: words can share distributional properties without being semantically similar. For example, antonyms often occur in similar contexts.

In this paper, we compare a prediction-based embedding model with a novel word embedding model, which purports to address the issues in the previous paragraph.

3 Data and Pre-processing

Both the models we discuss were trained on the English-French parallel EuroParl corpora (described in (Koehn, 2005)); Skipgram (SG) was trained on the English corpus, whilst Embed-Align (EA) – since it requires bilingual (aligned) training data – was trained on both corpora. For both models, stop words were removed using the NLTK toolkit.[1] We then replaced all the words that occurred only once in a corpus with the keyword `<unk>`.[2]

4 Models

4.1 Skipgram

We used Skipgram (SG) (introduced in (Mikolov et al., 2013)), an archetypal example of the prediction-based models described in the previous section. SG uses a simple feedforward neural network to approximate a function from words to their most likely contexts. A forward pass of the network is described as follows (where $\mathbf{i}_w$ is a one-hot vector indexing word w, W is a $|V| \times d$ matrix, where $|V|$ is the vocabulary size and d is the embedding dimension, and C is a $d \times |V|$ matrix):

$$\mathbf{h}_w = W\mathbf{i}_w \tag{1}$$

$$\mathbf{p}_w = softmax(C\mathbf{h}_w) \tag{2}$$

The word embedding for a word w is then its corresponding row in the matrix W.

We make use of `Word2Vec` function available in the gensim package (Řehůřek and Sojka, 2010). Rather than (expensively) computing the full $softmax$ at each forward pass, this implementation uses (Mikolov et al., 2013)'s negative sampling, updating only a subset of the model's weights for each training sample. The SG models are trained with multiple hyperparameter settings to assess their effect on the performance for each task. We report results for the best-performing SG model; its hyperparameters are outlined in Table 8. It is worth noting that we found that a context window $k = 3$ (rather than the more popular $k = 5$) worked best in practice.

4.2 Embed-Align

We used Embed-Align (introduced in (Rios et al., 2018)). We can see EA as being motivated by the problems for word embeddings raised in the previous section. In EA, word embeddings for a language of interest (here, English) are obtained by leveraging data from a parallel corpus in a second language (here, French) using a variational auto-encoder.

[1] `https://www.nltk.org`

[2] Since we used a provided pre-trained EA model, we attempted to mimic the pre-processing steps taken by Dr. Aziz in training SG, to facilitate comparison between the two models.

We refer the reader to (Rios et al., 2018) for the formal details of the model. Here, we content ourselves with describing the intuitive benefits of the EA framework.

In the EA model, there are two generative components, corresponding to categorical distributions over the vocabularies of each monolingual corpus. An inference model gives a (contextualized) location and scale parameter for each word in a given sentence in the language of interest; a reparameterization trick is used to ensure this posterior approximation is tractable (Kingma and Welling, 2013).

In other words, the inference model allows us to represent words as multivariate Gaussians, parametrized by location and scale vectors. The parametrization takes into account the contexts in which the words appear, allowing us to tease apart a given word's distinct semantic senses. Moreover, the fact that parallel alignment data is used (in the generative model) mitigates problems inherited from DH: since we are taking into account bilingual distributional properties (the properties of both a word and its foreign-language counterpart), there is less risk that two words will share distributional properties without being semantically similar.

5 Sentence Embeddings

We obtain sentence embeddings from our word embeddings by (a) averaging the embeddings of the individual words in the sentence (MEAN) (b) summing the embeddings of the individual words in the sentence (SUM) (c) weighted average of the embeddings as defined below (WEIGHTED),

$$\text{Sentence embedding} = \frac{\sum_i w_i x_i}{\sum_i w_i}$$

$$w_i = \frac{1}{\text{frequency of } w_i}$$

where w_i and x_i are the weight and embedding of the i^{th} word. The weight of each word is the inverse of its frequency in the corpora.

6 Evaluation and Results

SentEval is described in (Conneau and Kiela, 2018). Each of the constituent datasets is tokenized with the MOSES tokenizer (Koehn et al., 2007).

The SentEval tasks fall into two broad categories: 'Downstream Tasks', where sentence embeddings are indirectly evaluated by their performance on a high-level NLP task, and 'Probing Tasks', where sentence embeddings are evaluated more directly, by the information they contain about various grammatical features of the sentence. We present the results for each of these two types of task in thematic groups.

6.1 Sentiment analysis (binary classification)

We compare the models on a number of related binary classification tasks. There are three sentiment analysis tasks, in which the task is to classify the sentiment expressed by a given sentence as either positive or negative: the Movie Review (**MR**)[3] dataset contains $22k$ sentences, the Binary Sentiment Analysis (**SST**)[4] data set contains $70k$ sentences (both datasets are drawn from film reviews) and The Product Review (**CR**)[3] dataset contains $8k$ sentences drawn from product reviews.

There are also two binary classification tasks related to opinions. The Opinion Polarity (**MPQA**) dataset (Wiebe et al., 2005) contains $22k$ phrases expressing opinions; the task is to classify them as positive or negative. The Subjectivity Status (**SUBJ**) dataset contains $20k$ sentences drawn from film reviews; the task is to identify whether the sentence is expressing a fact or an opinion.

Each of these datasets was evenly divided into a test and training set. Since, for our purposes, this division is arbitrary (since no training occurs), we report the mean accuracy across these two sets. The results are presented in Table 1.

Model	MR	CR	SST	SUBJ	MPQA
EA (`SUM`)	66.71	72.14	68.08	83.36	83.87
EA (`MEAN`)	64.63	70.65	67.16	79.16	83.84
SG (`SUM`)	70.29	75.23	**74.15**	87.20	86.56
SG (`MEAN`)	**70.69**	**76.22**	73.61	**87.41**	**86.79**
SG (`WEI.`)	58.92	65.80	62.35	70.39	84.10

Table 1: Performance on each of the binary classification tasks. Each entry corresponds to mean accuracy across the training and test sets on the dataset.

Although SG performed better than EA on these tasks, both models performed significantly above random (50.00). Note that summing the EA word embeddings was more effective than averaging them. Whilst we obtain similar scores with EA (MEAN) to those reported in (Rios et al., 2018), EA (SUM) gets higher scores.

After we have touted the intuitive benefits of EA, it may disappoint the reader that SG outperforms the more advanced model on these tasks. One point which might help to explain EA's shortcomings here: a crucial benefit of the EA embeddings is that they are inherently contextualised. Words are represented depending on their sentential contexts. But the data upon which the models were trained is taken from a different context (a formal political environment) than the informal reviews which constitute these datasets. So, intuitively, we would not expect the learned contextualisation to have a pronouncedly positive effect here.

[3] `nlp.stanford.edu/~sidaw/home/projects:nbsvm`

[4] `nlp.stanford.edu/sentiment/treebank.html`

6.2 Multi-class Classification

The fine-grained Sentiment Analysis **(SST5)**[4] task uses a subset of the **SST** dataset (size $11k$), but with 5-way labels (corresponding to the strength of the positive sentiment). The Question-type Classification **(TREC)** dataset contains $6k$ questions; the task is to match each question with one of six types (Li and Roth, 2002). The results for these multi-class classification tasks are presented in Table 2. Again, both models perform significantly better than random. Again, note that EA (SUM) performs better than EA (MEAN) (especially on the TREC task), whilst SG (SUM) performs worse than SG (MEAN). As in the previous section, the differences between training data and test data could help explain why the intuitive benefits of EA did not manifest themselves.

Model	SST5	TREC
EA (SUM)	34.99	63.41
EA (MEAN)	33.83	55.11
SG (SUM)	38.60	76.56
SG (MEAN)	**38.76**	**77.79**
SG (WEIGHTED)	30.62	39.89

Table 2: Performance on each of the multi-class classification tasks.

6.3 Natural Language Inference

We compare the models on a natural language inference task and a paraphrase detection task.

The Sentences Involving Compositional Knowledge (SICK) dataset contains $10k$ pairs of English sentences, drawn from two larger video caption datasets (Marelli et al., 2014). In the SICK Entailment version of the dataset **(SICK-E)**, each pair is (manually) labelled with one of three semantic relations: entailment, contradiction or neutral (i.e. no relation). The data set contains 5595 neutral pairs, 1424 contradiction pairs, and 2821 entailment pairs.

The Microsoft Research Paraphrase Corpus **(MRPC)** contains $5.8k$ pairs of sentences extracted from web-based news sources (Dolan et al., 2004). Each sentence pair is accompanied by a binary label, indicating whether the sentences are semantically equivalent.

Model	SICK-E.	MRPC
EA (SUM)	74.30	74.30
EA (MEAN)	73.66	70.79
SG (SUM)	75.14	**76.34**
SG (MEAN)	**77.10**	76.15
SG (WEIGHTED)	60.72	65.45

Table 3: Performance on the Natural Language Inference and Paraphrase Detection tasks. Each entry corresponds to mean accuracy across the training and test sets on the dataset.

Again, both models perform significantly better than random, with SG giving a slight improvement on EA. We note once more that EA (SUM) yields better results than EA (MEAN).

6.4 Semantic Textual Similarity

We compare the models on Semantic Text Similarity (STS) task. Given a pair of sentences, STS measures the degree of equivalence in the semantics ranging from 0 to 5. A score of 0 indicates total independence and a score of 5 signifies equivalence. SentEval reports pearson and spearman correlation scores on Plagiarism detection, headlines, question-question, etc. We report only the weighted average of all the subtasks.

The STS 2016 **(STS16)** dataset consists of sentences from the 2016 SemEval task. The STS Benchmark **(STS-B)** dataset consists of sentences drawn from STS tasks for SemEval between 2012 and 2016.

In the SICK Relatedness version of the SICK dataset **(SICK-R)**, each pair is manually labelled with a score in the interval $[1, 5]$, corresponding to the degree of relatedness between the two sentences (Marelli et al., 2014). The data set contains 923 pairs in the range $[1, 2)$, 1373 pairs in $[2, 3)$, 3872 pairs in $[3, 4)$, and 3672 pairs in $[4, 5]$.

Model	STS-B	STS16	SICK-R.
EA (SUM)	0.61 0.60	**0.57** / **0.52**	0.62 / 0.67
EA (MEAN)	**0.61** / **0.61**	0.57 / 0.52	0.63 / 0.67
SG (SUM)	0.52 / 0.51	0.53 / 0.48	0.49 / **0.72**
SG (MEAN)	0.57 / 0.56	0.53 / 0.48	**0.66** / 0.71

Table 4: Performance on the Semantic Textual Similarity tasks. The first number in each entry is the spearman correlation and the second number corresponds to pearson correlation value.

6.5 Probing Tasks

The Probing tasks are introduced in (Conneau et al., 2018). All data sets contain $100k$ training instances, $10k$ development instances and $10k$ test instances. We report the mean accuracy on the validation and test sets.

6.5.1 Surface Information

The Length prediction **(SentLen)** task tests whether a sentence embedding contains information about its length. This is a 6-way classification task, where each category corresponds to a sentence length interval: (5-8), (9-12), (13-16), (17-20), (21-25), or (26-28).

Model	SentLen
EA (SUM)	73.95
EA (MEAN)	35.20
SG (SUM)	**74.01**
SG (MEAN)	51.25
SG (WEIGHTED)	23.50

Table 5: Results for the Length Prediction test.

As we would expect, summing the word embedding vectors produces a sentence embedding that is more sensitive to sentence length than taking the mean. It might seem surprising that the scores for the (MEAN) embeddings are as good as they are. This could be explained by the high-dimensionality of the embeddings; if individual word embeddings have dimensions with negligible components, then we would expect sentence length to be inferrable from the number of dimensions with components above some threshold (in other words, taking the mean is like summing for certain dimensions).

6.5.2 Syntactic Information

The Tree Depth Prediction **(TreeDepth)** task is to predict, given a sentence, the maximum depth of the sentence's syntactic tree. The task is 8-way classification, with depths between 5 and 12.

The Word order analysis **(BShift)** task tests whether a sentence embedding contains information about (legal) word order. For each well-formed sentence in the corpus, a dummy sentence is generated by randomly shuffling two adjacent words in the sentence. The task is then binary classification, namely to distinguish well-formed sentences from dummy sentences.

The Top Constituents prediction **(TopConst)** task tests whether a sentence embedding contains information about the high-level grammatical construction which formed it. The 19 most-common top constituent sequences in the corpus are identified (for example, 'NP-PP-.'). The task is then 20-way classification (the 20th category is a dummy category, designed to represent all constructions not in the 19 most common).

Model	TreeDepth	BShift	TopConst
EA (SUM)	28.40	**51.10**	35.40
EA (MEAN)	24.55	51.00	32.65
SG (SUM)	**31.99**	49.74	**64.85**
SG (MEAN)	31.89	49.71	64.10

Table 6: Syntactic Information

Note that the methods we use to generate sentence embeddings from word embeddings are insensitive to word order. That is, two different sequences of the same words will receive the same embeddings on our metrics. So we would expect all the models presented here to perform poorly on the **BShift** task. Since syntax tree depth correlates with sentence size, we would expect (SUM) to be more effective than (MEAN) on the **TreeDepth** task.

6.5.3 Semantic Information

The Verb tense prediction **(Tense)** task is a binary classification task, namely to predict (from the sentence embedding) whether the main verb in the sentence is past or present. Only high-frequency main verbs are used, and no verb appears in both the training and test set.

The Number prediction **SubjNum** (**ObjNum**) task is a binary classification task, namely to predict from the sentence embedding whether the subject (object) of the main verb in the sentence is singular or plural. Only high-frequency nouns are used, and no noun appears in both the training and test set.

The Semantic Odd Man Out **(SOMO)** task tests whether a sentence embedding contains contextual information about its constituent words. For each well-formed sentence in the corpus, a dummy sentence is generated by replacing a random word in the sentence with another word with the same Part of Speech (POS) tag. The task is then binary classification: is a given sentence semantically well-formed or not?

Our results for the semantic information tasks are presented in Table 7.[5]

Model	Tense	Subj	Obj	SOMO
EA (SUM)	71.30	73.85	69.75	49.75
EA (MEAN)	69.20	71.55	67.75	49.90
SG (SUM)	77.14	77.08	73.60	49.25
SG (MEAN)	**77.34**	**77.13**	**73.80**	49.20
SG (WEIGHTED)	69.15	64.05	61.20	**50.10**

Table 7: Semantic Information

7 General Discussion

Much of the discussion of results on individual tasks has taken place in the sections in which those results were presented. In this section, though, we focus on general insights that can be drawn from individual results.

As noted, distinct methods of composing sentence embeddings resulted in (statistically significant) differences in the performance of each model. The fact these differences exist is sufficient to motivate further work investigating more complex methods of composition.

For example, consider the difference between the two methods of generating sentence embeddings from word embeddings (summing or averaging) in the SICK-E task. The results for the SG models are presented in Figure 1. The significance of this result was checked us-

[5] Note that we do not present results from the Coordination Inversion task; since our sentence composition is order-insensitive, the models could not perform better than random.

ing a Bayesian paired T-Test. The Bayes factor quantifies the degree to which the data are more likely under one model versus another (Wagenmakers et al., 2018). Figure 2 shows the prior and posterior plot. The Bayes factor (~ 1016) reports a strong evidence towards the mean compositional method which supports our claim.

There is also an interesting discrepancy in these differences: the EA model performed better when individual word embeddings were summed, rather than averaged, whilst the SG model performed better when individual word embeddings were averaged.

This difference is not as surprising as it first appears. To see this, note that the contexts SG uses during training are fixed; the predictive task uses three context words on each side of a given word. With EA, though, the context is defined by the sentence boundaries. Moreover, as discussed, word alignment is a crucial component in the EA training. So, intuitively, we would expect a composition method which is sensitive to sentence length (such as SUM) to benefit EA more than SG.

8 Conclusion

In this paper, we have evaluated word embeddings from Skipgram and Embed-Align by their performance on a host of sentence-level tasks. We found that the embeddings from Skipgram performed better on the tasks than those from Embed-Align. More interestingly, we observed significant systematic differences in the effect the method of sentence composition had on the results for each model. In other words, we can (tentatively) conclude that a one-size-fits-all approach to composing word embeddings overlooks important idiosyncrasies of the individual word embedding models.

Future work is needed to support this conclusion. The sentence composition methods we considered here, though widely used, are rudimentary. For example, there were points at which the methods used to compose sentence embeddings from word embeddings hindered the models' ability to perform a given task (such as tasks involving word order). It would be interesting to examine the impact of more advanced word embedding composition techniques (such as Lexical Function models (Baroni and Zamparelli, 2010)) on each model's performance on the tasks.

A Code Repository

The code for the project can be found at the following URL: `https://github.com/Jack-Harding1/ULL`.

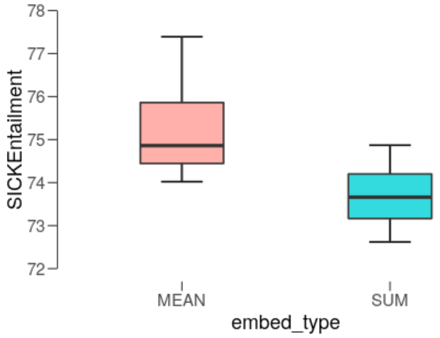

Figure 1: SG accuracy for the SICK-E task with MEAN and SUM sentence embeddings.

B Significance of MEAN vs SUM embedding

To test the significance of the mean and sum word embeddings, we conducted a Bayesian Paired T-Test (two-tailed). The Bayes factor compares the null hypothesis $\mathcal{H}_0$ (i.e., there is no effect resulting from the method of composition) to an alternative hypothesis $\mathcal{H}_1$ (i.e., there is an effect). The dataset contains accuracy for 40 SG models. Each data point is a pair of accuracies for 2 models with the only difference being the method of composition. The width of Cauchy prior, $r = 0.707$.

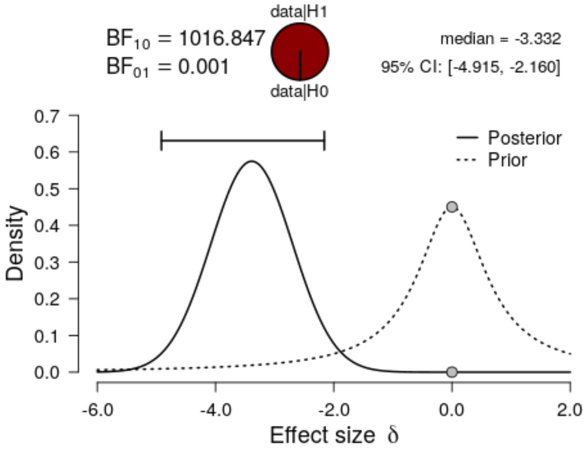

Figure 2: Prior-posterior plot for embedding type in SICK-Entailment task (Bayesian Paired T-Test).

C Hyperparameters

	value
Window length	3
Embedding size	200
Negative samples	5
Epochs	30

Table 8: Hyper parameters of the best SG model

References

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

Alexis Conneau and Douwe Kiela. 2018. Senteval: An evaluation toolkit for universal sentence representations. *arXiv:1803.05449v1*.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loc Barrault, and Marco Baroni. 2018. What you can cram into a single vector: Probing sentence embeddings for linguistic properties. *In Proceedings of ACL*.

B. Dolan, C. Quirk, and C. Brockett. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. *In Proceedings of ACL*.

Zellig S. Harris. 1954. Distributional structure. *Word*.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *https://arxiv.org/abs/1312.6114*.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. *MT Summit*.

Xin Li and Dan Roth. 2002. Learning question classifiers. *COLING'02*.

M. Marelli, S. Menini, M. Baroni, L. Bentivogli, R. Bernardi, and R. Zamparelli. 2014. A sick cure for the evaluation of compositional distributional semantic models. *Proceedings of LREC 2014, Reykjavik (Iceland): ELRA*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

George A. Miller and Walter G. Charles. 1991. Contextual correlates of semantic similarity. *Language and Cognitive Processes*.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May. ELRA. `http://is.muni.cz/publication/884893/en`.

Miguel Rios, Wilker Aziz, and Khalil Simaan. 2018. Deep generative model for joint alignment and word representation. *arXiv preprint arXiv:1802.05883*.

Wagenmakers, Eric-Jan, and al. 2018. Bayesian inference for psychology. part i: Theoretical advantages and practical ramifications. *Psychonomic Bulletin & Review*, Feb.

J. Wiebe, T. Wilson, and C. Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*.