Analysing Word Representation in the Input and Output Layers of Neural Language Models

Steven Derby **Paul Miller** **Barry Devereux**
Queen's University Belfast, Belfast, United Kingdom
{sderby02, p.miller, b.devereux}@qub.ac.uk

Abstract

Researchers have recently demonstrated that tying the neural weights between the input look-up table and the output classification layer can improve training and lower perplexity on sequence learning tasks such as language modelling. Such a procedure is possible due to the design of the softmax classification layer, which previous work has shown to comprise a viable set of semantic representations for the model vocabulary, and these these output embeddings are known to perform well on word similarity benchmarks. In this paper, we make meaningful comparisons between the input and output embeddings and other SOTA distributional models to gain a better understanding of the types of information they represent. We also construct a new set of word embeddings using the output embeddings to create locally-optimal approximations for the intermediate representations from the language model. These locally-optimal embeddings demonstrate excellent performance across all our evaluations.

1 Introduction

Neural Language Modelling has recently gained popularity in NLP. A Neural Network Language Model (NNLM) is tasked with learning a conditional probability distribution over the occurrences of words in text (Mikolov et al., 2011). This language modelling objective requires a neural network with sufficient capacity to learn meaningful linguistic information such as semantic knowledge and syntactic structure. Due to their ability to learn these important linguistic phenomena, NNLMs have been successfully employed as an effective method for generative pretraining (Dai and Le, 2015) and transfer learning to other natural language tasks (Peters et al., 2018a; Howard and Ruder, 2018; Radford et al., 2018). As previously suggested by Bengio et al. (2003), Mnih and Hinton (2007) and Mnih and Teh (2012), the weights of the final fully-connected output layer, or output embeddings, which compute the conditional probability distribution over the lexicon, also constitute a legitimate set of embedding vectors representing word meaning, as is the case for the input embeddings. This commonality between the input and output layers of the NNLM has motivated researchers to tie these representations together during training, improving performance on language modelling tasks (Inan et al., 2016; Press and Wolf, 2017). Furthermore, such a procedure is intuitive, since both the input and output embeddings of the network would appear to be performing a similar task of encoding information about lexical content. As described by Inan et al. (2016), they clearly live in an identical semantic space in language models, unlike other machine learning models were the input and output embeddings have no direct link.

On the other hand, it would also be reasonable to assume that the output representations require highly task-specific features (Peters et al., 2018a,b; Devlin et al., 2019). Despite their utility in language modelling, in-depth analysis of these input and output vector representations remains limited. The goal of this work is to gain a deeper understanding of the aspects of language captured in these contrasting representations. Our two main contributions[1] are as follows:

1. We perform an investigation to uncover both the broad types of semantic knowledge and fine-grained linguistic phenomena encoded within each set of word representations.
2. We propose a simple method for constructing locally-optimal approximations that we use to extend our analysis to the intermediate representations from the network.

[1] Code available at https://github.com/stevend94/CoNLL2020

Proceedings of the 24th Conference on Computational Natural Language Learning, pages 442–454
Online, November 19-20, 2020. ©2020 Association for Computational Linguistics
https://doi.org/10.18653/v1/P17

Though generally considered task-agnostic, by making extensive comparisons between these neural representations we may reason about the type of information most salient in the representations in each semantic space. Our results demonstrate that the input and output embeddings share little in common with respect to their strength and weaknesses, while the locally-optimal embeddings generally perform the best on most downstream tasks.

2 Related Work

Recent trends in NLP has seen a focus towards building generative pretraining models, which have achieved state-of-the-art performance on downstream tasks (Peters et al., 2018a; Radford et al., 2018; Devlin et al., 2019; Lan et al., 2019; Liu et al., 2019; Yang et al., 2019). These sequence-based autoencoder models have almost universally adopted the convention of weight tying in their input and output layers, which has been shown to improve training and decrease perplexity scores on language modelling tasks (Inan et al., 2016; Press and Wolf, 2017). Motivated by these results, researchers have proposed a number of modifications to these networks in relation to the output classification layers. For example, Gulordava et al. (2018a) combine weight-tying with a linear projection layer in the penultimate stage of the network to both decouple hidden state representations from the output embeddings and control the size of the embedding vectors. Takase et al. (2017) suggest modifying the architecture of the network by adding a gating mechanism between the input layer and the final classification layer of NNLMs. Focusing solely on the final classification layer, Yang et al. (2017) propose using a number of weighted softmax distributions, called a *Mixture of Softmaxes*, to overcome the bottleneck formed by their limited capacity. Takase et al. (2018) extend this approach by adding what they call a *Direct Output Connection*, which computes the probability distribution at all layers of the NNLM. Other work has focused on weight tying such as with the *Structural Aware* output layer (Pappas et al., 2018; Pappas and Henderson, 2019). Despite their importance, there is limited work which attempts to further analyse these output embeddings beyond the work of Press and Wolf (2017), who show that these representations outperform the input embeddings on word similarity benchmarks. In recent years, such analyses has gained popularity in the NLP community as researchers have shifted their focus towards interpretability in neural networks (Alishahi et al., 2019; Linzen et al., 2019). Examples include probing tasks, which are supervised machine learning problems that look to decode salient linguistic features from embedding vectors (Adi et al., 2016; Wallace et al., 2019; Tenney et al., 2019). Other work has focused on determining whether more cognitive aspects of meaning are adequetely encoded within these representations, through probing (Collell and Moens, 2016; Li and Gauthier, 2017; Derby et al., 2020) or using cross-modal mappings (Rubinstein et al., 2015; Fagarasan et al., 2015; Bulat et al., 2016; Derby et al., 2019; Li and Summers-Stay, 2019). Moving beyond basic linguistic phenomena, researchers have also investigated more complex aspects of language such as syntactical structure using probing methods (Linzen et al., 2016; Bernardy and Lappin, 2017; Gulordava et al., 2018b; Marvin and Linzen, 2018).

3 Research Context and Motivation

In this section, we first discuss some background about the input and output embeddings in NNLMs. Then, we briefly discuss how to compute new representations that are locally-optimal to the prediction step from the fully-connected softmax layer of the NNLM, by using stochastic gradient descent.

3.1 Neural Network Language Model

Consider a sequence of text $(y_1, y_2, \dots y_N)$ represented as a list of one-hot token vectors. The goal of a neural network language model is to maximize the conditional probability of the next word based on the previous context. For a vocabulary V, at the time step $t-1$ the network computes the probability distribution y^*_t of possible target words as follows:

$$\begin{aligned} e_t &= Ey_{t-1} \\ h_t &= f(e_t, h_{t-1}) \\ a_t &= Wh_t + b \\ y^*_t &= \text{Softmax}(a_t) \end{aligned} \tag{3.1}$$

where f consists of one or many temporally compatible layers, such as LSTMs (Hochreiter and Schmidhuber, 1997) or masked transformers (Vaswani et al., 2017). The function f takes in a previous state as contextual information $h_{t-1} \in \mathbb{R}^{d_f}$ and embeddings e_t from the look-up table $E \in \mathbb{R}^{d_e \times |V|}$, and produces a new hidden state h_t which the fully-connected output layer uses to compute the probability distribution y_t^*. We then compute the cross-entropy loss $\mathcal{L}(y_t, y_t^*)$ between the predicted distribution and the actual distribution, and minimize the loss with gradient descent.

To consider the case of weight-tying, we first note the fact that the size of the predicted probability distribution must span the length of the lexicon V. Then, disregarding the bias term, as $W \in \mathbb{R}^{|V| \times d_f}$ it is easy to see how we can set $E = W^T$ if we set $d_f = d_e$. Weight tying has several advantages, including less training parameters and improved perplexity scores on language modelling objectives (Inan et al., 2016; Press and Wolf, 2017). However, the information that both the input and output embeddings must individually learn in order to predict the correct target concept may be entirely different.

3.2 Hidden State Word Representations

While these output embeddings can function as a set of semantic representations, their real goal is to instead compute the conditional probability distribution over the lexicon using context information from the hidden layers of the network. As such, the output embeddings may contain certain features that are specific to the language modelling objective, allowing them to identify information from the hidden layers that is relevant to predicting the target word. In addition to considering the input and output embeddings, we also consider the activation vectors from the latent layers of the language model in order to extend the scope of our analysis. From the perspective of how these layers represent lexical information, we are interested in the activation vectors in the hidden layers that lead to high prediction probabilities for the target words.

Intuitively, in order to find some activation vector from the latent layers that best represents a particular word, we would like to generate a sentence fragment that is optimal with respect to predicting that word (i.e. the hidden state h_t for the sentence fragment yields the highest possible probability value for the target word being the next word in the sequence, given the calculations in Eqn. 3.1). We could then use these hidden state activations for each word as an additional embedding space, similar to Bommasani et al. (2020). However, we lack an efficient generative process for finding such optimal sentence fragments. We could sample a large number of sentence fragments from a corpus and record which sentence fragments give the highest output probability for each word in our lexicon, but this will be highly inefficient and moreover will not guarantee that we have found the *best* hidden state activation vector for each word.

In the next section, we present a procedure to identify such optimal hidden states, which we refer to as *locally-optimal vectors*.

3.3 Locally-optimal Vectors

To find a latent representation that maximally predicts the target word from the final classification layer of the NNLM, we build a gradient-based approximation for each word. To achieve this, we employ a similar technique to *Activation Maximization* in computer vision (Simonyan et al., 2013). For a pretrained NNLM, let $W \in \mathbb{R}^{d_f \times |V|}$ be the weight matrix (i.e. output embeddings) and let $b \in \mathbb{R}^{|V|}$ be the bias vector of the final prediction layer of the network. For each word in $w \in V$, we want to find the corresponding input $I \in \mathbb{R}^{d_f}$ that maximizes the probability of the word w. Let S_w be the score function for the word $w \in V$, which takes an input and gives the probability output of the target class w. We can then formulate the problem as

$$\arg\max_{I \in \mathbb{R}^{d_f}} S_w(I) - \lambda ||I||_2^2 \tag{3.2}$$

where

$$S_w(I) = \text{Softmax}(W^T I + b)_w \tag{3.3}$$

where λ is a regularisation parameter. As described by Simonyan et al. (2013), maximizing the class probability can be achieved by minimizing the score for incorrect classes. This is undesirable for visualization purposes (see Simonyan et al., 2013), which is the reason why softmax normalization is usually omitted, though in our case, finding the most probable class is desirable. The regularisation term stops the magnitude of the vectors growing too large and instead focuses on the angular information between representations. We refer to these representations as *AM Embeddings*. Although these embeddings have the same dimensionality as the hidden states h_t in the NNLM and play the same role in the softmax calculation, we note that they are not derived from any particular text sequence input to the NNLM and indeed there may not exist any sentence fragment that produces these hidden state activations.

4 Methodology

For our research, we require a NNLM that provides good performance without weight tying, so that

	Semantic Similarity		Semantic Relatedness			Hybrid		BrainBench	
Models	WordSim-S	SimLex999	WordSim-R	MEN	MTurk	WordSim	RW	fMRI	MEG
Distributional Semantic Models									
Word2Vec	0.759	0.400	0.555	0.725	0.660	0.645	0.637	0.687	0.677
GloVe	0.680	0.352	0.475	0.727	0.604	0.546	0.530	0.657	0.623
FastText	0.782	0.391	0.585	**0.742**	0.678	0.668	0.647	0.680	0.682
Pretrained NNLM Representations									
Input Embs.	0.734	0.420	0.361	0.640	0.556	0.527	**0.694**	0.661	0.683
Output Embs.	0.771	0.417	0.543	0.677	0.642	0.635	0.541	0.699	0.709
AM Embs.	**0.793**	**0.486**	**0.614**	0.741	**0.685**	**0.692**	0.649	**0.705**	**0.710**

Table 1: Results (accuracy % for BrainBench; Spearman's ρ for the other evaluations) for the three JLM-derived embedding spaces, along with three other state-of-the-art distributional semantic models.

we may analyse the input and output embeddings as separate entities. We use the freely-available language model of Jozefowicz et al. (2016), which we refer to as **JLM**. The JLM network consists of a character level embedding input and two LSTM layers of size 8192, which both incorporate a projection layer to reduce the hidden state dimensionality down to 1024. The softmax output of the model has a word-level vocabulary of 800K word classes, and the model is trained on the one billion word news dataset (Chelba et al., 2013).

4.1 Pretrained NNLM Embeddings

We first acquire the input and output embeddings by extracting the appropriate matrices from their respective locations in the JLM network, with the input embeddings generated using the character-level layers. We then construct the AM embeddings, first by randomly initialising a set of $|V|$ vectors before optimising using the Adam optimiser with a learning rate of 0.001 and regularisation term λ=10^{-5}. We train for 100 epochs, with a batch size of 1024 using *Keras*. Due to the enormous size of the lexicon of the JLM language model, we downsample the 800K word vocabulary by taking the first 20K most frequently occurring words, which gives good coverage over the evaluation datasets.

4.2 Distributional Semantic Models

We also want to compare these embeddings with state-of-the-art distributional semantic models in order to make meaningful comparisons. For this, we use the skip-gram implementation of *Word2Vec* (Mikolov et al., 2013) and *FastText* (Bojanowski et al., 2017) using the *gensim* package[2] and the Python implementation of Facebook's *FastText*[3] respectively. *Word2Vec* was trained with embeddings of size 300 and a context window of 5, while *FastText* uses the default settings with embedding size 100, window size 5, and ngrams of sizes from 3 to 6. We also train a Python implementation of *GloVe* (Pennington et al., 2014) for 100 epochs with a learning rate of 0.05 to construct word embeddings of size 300. For a fair comparison, all models are trained on the same billion-word dataset (Chelba et al., 2013) as JLM.

5 Experiments

To assess these representations for both task-specific effectiveness and fine-grained linguistic knowledge, we perform a broad range of experiments. These assessments include comparison with human understanding on word relations (Intrinsic Evaluations), analysing performance on supervised machine learning tasks (Extrinsic Evaluations), and using probing tasks to isolate linguistic phenomena. We hypothesise that the input and output embeddings should perform quite well on the intrinsic benchmarks, while the AM embeddings should give the best results on downstream prediction tasks, which we would similarly expect with the hidden representations from the intermediate layers of the network (Peters et al., 2018a).

5.1 Intrinsic Evaluations

We first compare the word embeddings with human semantic judgements of word pair similarity. The rationale is that a good semantic model should correlate with semantic ground-truth information elicited from humans, either from conscious judgments, or from patterns of brain activation as people process the words (Bakarov, 2018).

[2] https://radimrehurek.com/gensim/

[3] https://pypi.org/project/fasttext/

	Binary Classification					Multiclass		Entailment	Paraphrase
Models	MR	CR	MPQA	Subj.	SST2	SST5	TREC	SICK-E	MRPC
Distributional Semantic Models									
Word2Vec	70.76	71.18	85.88	86.34	75.78	38.82	79.20	71.38	66.49 / 79.87
Glove	68.22	69.59	84.58	86.94	73.86	36.15	78.60	70.83	69.57 / 80.97
FastText	70.64	67.87	85.77	87.59	77.59	38.87	73.20	70.57	66.49 / 79.87
Pretrained NNLM Representations									
Input Embs.	71.32	**76.56**	87.04	88.45	76.39	40.05	85.6	**79.58**	**73.22 / 81.51**
Output Embs.	72.10	67.13	87.57	88.37	79.24	39.37	81.60	75.44	68.58 / 80.82
AM Embs.	**72.76**	75.15	**87.76**	**89.20**	**79.85**	**41.27**	**86.00**	75.14	66.49 / 79.87

Table 2: Results on the *SentEval* transfer learning tasks measured in % accuracy for all six of our embeddings. Each task is grouped into four categories, *Binary Classification*, *Multiclass Classification*, *Entailment/Relatedness* and *Paraphrase Detection*. For the *MRPC* dataset, the results are % accuracy and F1 × 100.

Similarity Benchmarks A traditional method for evaluating word embeddings uses the intuition of human raters about word semantic similarity. Word similarity benchmarks can, in general, be partitioned into two types: *semantic similarity* and *semantic relatedness*. Here, semantic relatedness refers to the strength of association between words (e.g. COFFEE and CUP), while semantic similarity reflects shared semantic properties (e.g. COFFEE and TEA). For benchmarks focusing on semantic relatedness/association, we use **MEN** (Bruni et al., 2012), **MTurk** (Radinsky et al., 2011) and **WordSim353-Rel** (Agirre et al., 2009), and for semantic similarity we use **SimLex-999** (Hill et al., 2015), and **WordSim353-Sim** (Agirre et al., 2009). We also include two datasets whose judgement scores do not fall into either category, **WordSim353** (Finkelstein et al., 2002) and **RareWords** (Luong et al., 2013). For the embedding vectors, similarity is computed using the cosine between pairs of word vectors, with Spearman's ρ used to measure the correlation between human scores and the cosine similarities. We perform our analysis using the *Vecto* python package (Rogers et al., 2018)[4].

Predicting Brain Data We also evaluate these embeddings on another intrinsic evaluation task that does not directly employ human semantic judgement. Instead, this evaluation asks whether the embedding models can reliably predict activation patterns in human brain imaging data as participants processed the meanings of words. For this, we use **BrainBench** (Xu et al., 2016)[5], a semantic evaluation platform that includes two separate neuroimaging datasets (fMRI and MEG) from humans for 60 concept words. This benchmark evaluates how well the embeddings can make predictions about the neuroimaging data using a 2 vs. 2 test, with 50% indicating chance accuracy.

Intrinsic evaluation results In general, the output embeddings perform better than the input embeddings (Table 1), similar to (Press and Wolf, 2017). The only case where the input embeddings yield higher correlations than the output embeddings are on *Rare Words*. We can attribute this to the fact that the input embeddings are constructed from character-level representations. In comparison to the SOTA distributional models, the output embeddings tend to only beat *FastText* on *SimLex999* and *BrainBench*, while also struggling in comparison to *Word2Vec* on semantic relatedness and hybrid tasks. On the other hand, our AM embeddings perform very well in all evaluations, being the top-preforming model in most evaluations and performing quite similarly to *FastText* on MEN and *Rare Words*. While we hypothesised that the AM embeddings should perform quite well on downstream tasks, the ability of these novel word embeddings to explain human semantic judgement and reliably decode brain imaging data is surprising and interesting.

5.2 Extrinsic Evaluations

Next, we evaluate these representations by analysing their performance on a number of downstream tasks. Each task may demand a certain set of features relevant to the task, requiring these representations to encode a wide range of linguistic knowledge. We expect the output embeddings to perform better than the input embeddings and other

[4] https://vecto.readthedocs.io

[5] http://www.langlearnlab.cs.uvic.ca/brainbench/

Models	SICK-R	STS B
Distributional Semantic Models		
Word2Vec	75.32	57.93
Glove	71.64	56.03
FastText	75.95	58.52
Pretrained NNLM Representaitons		
Input Embs.	**79.49**	**62.23**
Output Embs.	78.86	61.72
AM Embs.	78.69	61.13

Table 3: Pearson correlation results on SICK-R and STS B semantic relatedness benchmarks.

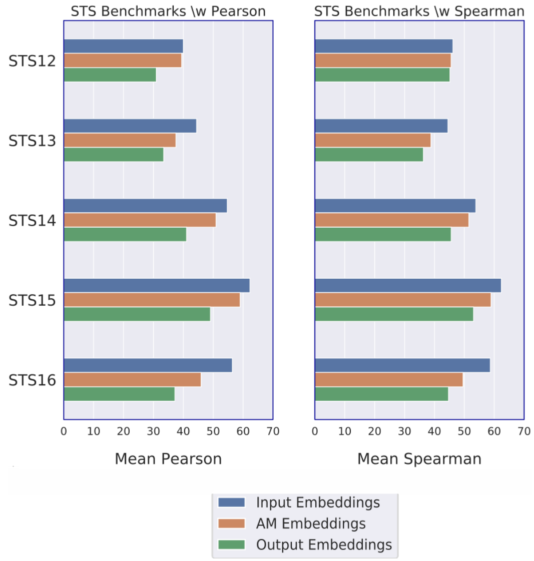

Figure 1: Results on STS benchmarks from *SentEval* toolkit. Here we report average *Pearson* and *Spearman* correlation scores on each benchmark.

SOTA semantic models based on previous research, which demonstrates that representations from the upper layers of the NNLM tend to perform better at prediction tasks (Peters et al., 2018a,b; Devlin et al., 2019). Since the AM embeddings represent a locally-optimal instance for the penultimate layer of the network, we also expect them to perform well.

Transfer Learning Tasks We make use of *SentEval* (Conneau et al., 2017), an evaluation suite for analysing the performance of sentence representations. Though we are working with word embeddings, applications rarely require words in isolation. To build sentence embeddings, we take the average embedding vector of all words in the sentence. *SentEval* includes a number of binary classification datasets, including two movie review sentiment datasets (**MR**) (Pang and Lee, 2005) and (**SST2**) (Socher et al., 2013), a product review dataset (**CR**) (Hu and Liu, 2004), subjectivity dataset (**Subj.**) (Pang and Lee, 2004) and an opinion polarity dataset (**MPQA**) (Wiebe et al., 2005). It also includes two multiclass classifications tasks, a question type classification dataset (**TREC**) (Voorhees and Tice, 2000) and a movie review dataset with five sentiment classes (Socher et al., 2013), as well as an entailment dataset (**SICK-E**) (Marelli et al., 2014) and paraphrase detection dataset (**MRPC**) (Dolan et al., 2004). For classification, we use a one-layer PyTorch GPU model with default parameters and Adam optimisation.

The results (Table 2) show that, on binary classification tasks, the input and output embeddings perform quite similarly, while both provide better results than the distributional models in almost all cases. Taking a closer look, we can see that the output embeddings perform best at predicting movie review sentiment (**MR**, **SST2**) and opinion polarity (**MPQA**), while the input embeddings provide the highest scores when predicting product review sentiment (**CR**) and subjectivity (**Subj.**). When predicting multiple classes (**TREC**, **SST5**), the input embeddings perform marginally better than the output embeddings, though the AM embeddings perform best overall on both binary and multiclass datasets. Interestingly, the input embeddings are much better at both predicting entailment (**SICK-E**) and paraphrase detection (**MRPC**) than all other models.

Semantic Text Similarity To further evaluate how well these embeddings perform at judging sentence relations, we also employ transfer learning to the semantic relatedness tasks from *SemEval*, in particular **SICK-R** (Marelli et al., 2014) and **STS B** (Cer et al., 2017). The task consists of sentence pairs with scores ranging from 0 to 5, indicating the level of similarity between the sentences. We see from the results (Table 3) that the input embeddings again give the highest correlation with semantic relatedness scores, similar to the previous results. Furthermore, the AM embeddings perform worse at judging relatedness than the output embeddings, though the differences are quite small. Our AM embeddings still outperform all SOTA distributional models.

We also perform transfer learning on a set of Semantic Textual Similarity (**STS**) benchmarks,

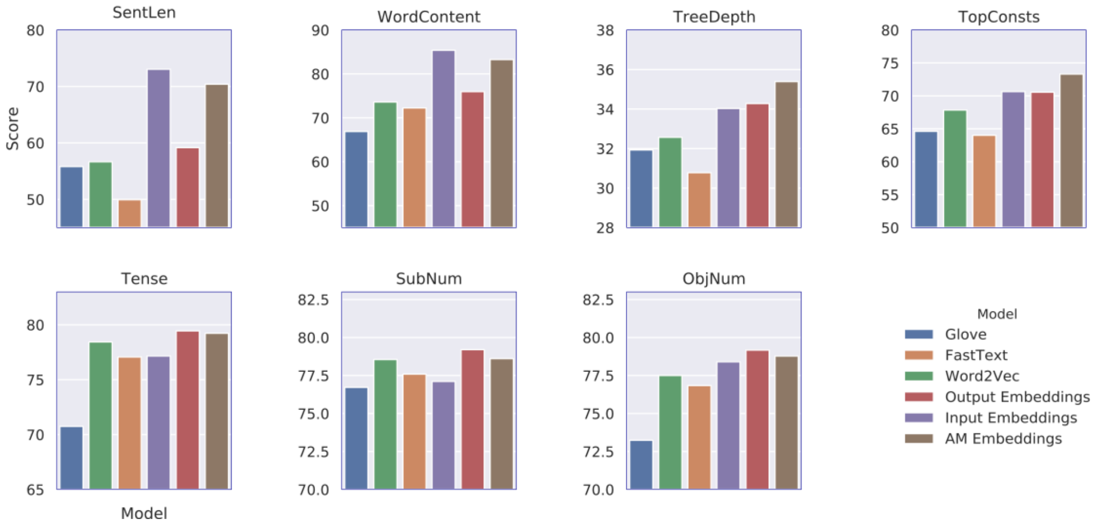

Figure 2: Results on each probing task, divided into three categories of task; **Surface Information**, **Syntactic Structure** and **Semantic Information**.

which include the 2012 (Agirre et al., 2012), 2013 (Agirre et al., 2013), 2014 (Agirre et al., 2014), 2015 (Agirre et al., 2014) and 2016 (Agirre et al., 2016) semantic similarity tasks. Each dataset contains sentence pairs similar to the relatedness tasks, though each is taken from different sources such as news articles or forums. Here, we record performance using the average *Pearson* and *Spearman* correlation for each *STS* dataset, with results displayed in Figure 1. The input embeddings again give the best performance on all datasets, similar to previous results on sentence relatedness. Furthermore, the AM embeddings perform better than the output embeddings on all datasets, in contrast to the previous findings. The results demonstrate that the input embeddings are much more suited to sentence comparison tasks than the other pretrained NNLM embeddings.

5.3 Probing Tasks

We next examine whether the embedding vectors capture certain linguistic properties when utilised as sentence representations. These probing tasks are formulated as a supervised classification problem, with strong performance indicating the presence of an isolated characteristic such as sentence length. Similar to the transfer learning tasks, we take the average embedding vector of all words to generate the sentence embedding. These tasks are taken from Conneau et al. (2018), which includes probing tasks partitioned into three separate categories.

- **Surface Information**: The tasks include sentence length prediction (**SentLen**) and deciding whether a word is present in the representations (**WordContent**).
- **Syntactical Information**: Focusing on grammatical structure, these include tasks for predicting the maximum length of a node to the root (**TreeDepth**) and predicting the top constituent below the <S> node (**TopConsts**).
- **Semantic Information**: Focusing on dependency knowledge, these include tasks for predicting the tense of the main verb (**Tense**), the number of subjects of the main clause (**SubjNum**) and the number of objects of the main clause (**ObjNum**).

We exclude other probing tasks that rely on word position in the sentence, since these averaged word embeddings are invariant with respect to word order[6]. The results are displayed in Figure 2. The SOTA distributional models tend to perform worse than the pretrained NNLM representations when predicting **SentLen** and **WordContent**, though the output models perform poorly compared to the input and AM embeddings. The AM embeddings perform well, perhaps because of their training objective which incentivises linear separability. When predicting syntactic information, the input and output embeddings perform similarly at classifying **TreeDepth** and **TopConsts**, with the AM embeddings performing best. Finally, when predicting

[6] Results on these tasks confirm this, with accuracy at chance levels.

	NNLM$_{\mathbf{Input}}$			NNLM$_{\mathbf{Output}}$			NNLM$_{\mathbf{Tied}}$		
Models	Train	Validation	Test	Train	Validation	Test	Train	Validation	Test
Distributional Semantic Models									
Word2Vec	57.0	58.9	59.3	59.0	67.2	67.4	74.9	64.9	65.3
Glove	61.8	62.5	63.1	66.3	74.4	74.5	90.5	76.8	77.4
FastText	58.0	59.7	60.2	60.6	68.9	69.0	77.9	67.6	68.0
Pretrained NNLM Representaitons									
Inputs Embs.	51.1	57.0	57.2	58.6	66.0	66.23	72.4	64.9	65.2
Output Embs.	52.4	57.6	58.1	**47.9**	**56.1**	**56.0**	55.5	53.3	53.3
AM Embs.	**48.7**	**55.8**	**56.40**	48.5	56.50	56.7	**52.7**	**52.0**	**52.4**

Table 4: Perplexity scores on the Penn Treebank for language models trained using each embedding model as fixed vector inputs, fixed weight outputs or both tied together.

Tense, **SubjNum** and **ObjNum**, the output embeddings are superior, which may be due to the output embeddings heavily encoding dependency information that is relevant to predicting the upcoming word during language modelling. Indeed, LSTMs are particularly good at learning dependency information such as subject-verb agreement (Linzen et al., 2016).

6 Neural Language Modelling

We have demonstrated that the linguistic knowledge captured by the input and output embeddings are moderately distinct. These results may imply that the input and output embeddings of the NNLM require a particular set of non-overlapping characteristics that are important to their respective roles in the NNLM. To further understand whether and how these representations are distinctive to their particular functions in the input and output layers, we perform domain transfer on the language modelling objective. For our evaluation, we test each set of embedding vectors when fixed as certain weights in the network:

1. **NNLM$_{\mathbf{In}}$**: Fixing our embedding vectors as the lookup table input to the language model.
2. **NNLM$_{\mathbf{Out}}$**: Fixing the softmax output layer by using the transpose of the stacked embedding vectors as the matrix of dense weights, without a bias vector.
3. **NNLM$_{\mathbf{Tied}}$**: Fixing the embedding inputs and softmax output by using our embeddings as the tied weights.

Here we expect the input embeddings and output embeddings to perform well in the case of **NNLM$_{\mathbf{In}}$** and **NNLM$_{\mathbf{Out}}$** respectively, since in these cases their role is congruent with their original role in JLM. We also expect the other distributional models to perform well as input embeddings based on previous research. It will also be interesting to see how the AM representations perform since they are trained using output embeddings and thus should share a lot of their linguistic knowledge. If the input and output embeddings perform similarly, we can infer that these representations contain considerable overlap in lexical information. However, if they perform poorly when their roles are switched, we can conclude that these representations must learn some role-specific features not encoded in the other semantic spaces. See the appendix for training details, which closely follow the medium-sized LSTM model presented by Zaremba et al. (2014) with the Penn Treebank dataset (Marcus et al., 1993).

6.1 Perplexity Results

Results are displayed in Table 4. In the **NNLM$_{\mathbf{In}}$** models, we see that the AM embeddings provide the best performance, even outperforming the input embeddings, with the output embeddings and SOTA distributional models performing quite well. We also note that the input embeddings still provide slightly better performance than the output embeddings in this analysis. In the case of the **NNLM$_{\mathbf{Out}}$** networks, most of the distributional models perform poorly. The NNLM struggles when the distributional models are utilised as fully-connected classification weights, while the output embeddings, which were trained for this task, perform best, though the AM embeddings also perform well. The input embeddings perform poorly in the **NNLM$_{\mathbf{Out}}$** model, indicating that the output embeddings do encode role-specific knowledge not captured by the other distributional models. Finally, when we tie and fix the weights, the SOTA distributional models and input embeddings do not improve the performance much in the **NNLM$_{\textbf{Tied}}$** model. Both the output embeddings and AM embeddings have good performance, and our AM embeddings surprisingly give the best results.

7 Discussion

We can draw several conclusions from these results. As expected, the type of semantic knowledge these representations capture is dependent on their position in the network.

7.1 Semantic Knowledge

The input embeddings struggle with representing word-level semantic relationships though perform well at estimating relatedness between sentences and paraphrase detection. The input embeddings also seem to encode several aspects of surface-level information such as sentence length, which is behavior more expected of contextualised representations of meaning. Indeed, the input embeddings seem to contain at least some qualities that make them suitable for building sentence-level representations. On the other hand, the output embeddings struggle as sentence-level representations. This is not so surprising, since these embeddings are the input components used to construct contextual representations in the intermediate layers, unlike the output embeddings.

The output embeddings seem to correlate more closely with human judgment on the word-level association and neuroimaging data for isolated concept words than the input embeddings. Furthermore, the output embeddings are highly task-specific to language modelling. Though other distributional semantic models estimate representations of meaning through somewhat similar language modelling objectives, they fail to learn any meaningful knowledge that is transferable to the output classification layer of the language modelling task.

7.2 Weight Tying

There are a number of characteristics that each set of representations seem to capture quite well given their position in the architecture of the NNLM. In a tied representation, we would expect the network to learn a set of embedding vectors that encode all such knowledge, though the contribution from each layer may not be entirely equal. Press and Wolf (2017) noted that, due to the update rules that occur when using weight tying between these layers, the output embeddings get updated at each row after every iteration, unlike the input embeddings. This implies a greater degree of similarity of the tied embedding to the untied model's output embedding than to its input embedding. From the perspective of this work, we would also add that a tied representation would be more similar to the output embeddings since the information they capture is more important to the overall learning objective. Based on our results, while the output embedding knowledge is quite transferable to the input embeddings, the converse is false.

7.3 Transfer Learning

In recent years, representations from pretrained neural language models have become a popular choice for transfer learning to other tasks. Generally, the intermediate representations from the layers of the network are preferred, since they are contextualised over the sentence and generally perform better in downstream tasks. In our work, we use the AM embeddings to behave as a stand-in for the intermediate layers' hidden states that are locally-optimal to each particular target word. Similar to these intermediate representations, our AM embeddings perform quite well on downstream NLP tasks. While this is to be expected, the results on the intrinsic evaluations and language modelling tasks are surprising. We would expect these embeddings to learn quite a bit of knowledge from the output embeddings, though the increase in performance on some tasks is striking. This may be due to the activation maximisation training objective that we employ, which forces linear separability between words in the lexicon whilst preserving the semantic information about each word (see Appendix).

8 Conclusion

We perform an in-depth analysis of the input and output embeddings of neural network language models to investigate what linguistic features are encoded in each semantic space. We also extend our analysis by constructing locally-optimal vectors from the output embeddings, which seem to provide overall better performance on both intrinsic and extrinsic evaluation tasks, beating well-established distributional semantic models in almost all evaluations.

References

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2016. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. *arXiv preprint arXiv:1608.04207*.

Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Paşca, and Aitor Soroa. 2009. A study on similarity and relatedness using distributional and WordNet-based approaches. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 19–27, Boulder, Colorado. Association for Computational Linguistics.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2014. SemEval-2014 task 10: Multilingual semantic textual similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 81–91, Dublin, Ireland. Association for Computational Linguistics.

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2016. SemEval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 497–511, San Diego, California. Association for Computational Linguistics.

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. SemEval-2012 task 6: A pilot on semantic textual similarity. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 385–393, Montréal, Canada. Association for Computational Linguistics.

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. *SEM 2013 shared task: Semantic textual similarity. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pages 32–43, Atlanta, Georgia, USA. Association for Computational Linguistics.

Afra Alishahi, Grzegorz Chrupała, and Tal Linzen. 2019. Analyzing and interpreting neural networks for NLP: A report on the first blackboxnlp workshop. *Natural Language Engineering*, 25(4):543–557.

Amir Bakarov. 2018. A survey of word embeddings evaluation methods. *arXiv preprint arXiv:1801.09536*.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155.

Jean-Philippe Bernardy and Shalom Lappin. 2017. Using deep neural networks to learn syntactic agreement. *LiLT (Linguistic Issues in Language Technology)*, 15.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Rishi Bommasani, Kelly Davis, and Claire Cardie. 2020. Interpreting Pretrained Contextualized Representations via Reductions to Static Embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4758–4781, Online. Association for Computational Linguistics.

Elia Bruni, Gemma Boleda, Marco Baroni, and Nam-Khanh Tran. 2012. Distributional semantics in technicolor. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 136–145, Jeju Island, Korea. Association for Computational Linguistics.

Luana Bulat, Douwe Kiela, and Stephen Clark. 2016. Vision and feature norms: Improving automatic feature norm learning through cross-modal maps. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 579–588.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2013. One billion word benchmark for measuring progress in statistical language modeling. *arXiv preprint arXiv:1312.3005*.

Guillem Collell and Marie-Francine Moens. 2016. Is an image worth more than a thousand words? on the fine-grain semantic differences between visual and linguistic representations. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2807–2817. The COLING 2016 Organizing Committee.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark. Association for Computational Linguistics.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136, Melbourne, Australia. Association for Computational Linguistics.

Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in neural information processing systems 28*, pages 3079–3087. Curran Associates, Inc.

Steven Derby, Paul Miller, and Barry Devereux. 2019. Feature2Vec: Distributional semantic modelling of human property knowledge. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5853–5859, Hong Kong, China. Association for Computational Linguistics.

Steven Derby, Paul Miller, and Barry Devereux. 2020. Encoding lexico-semantic knowledge using ensembles of feature maps from deep convolutional neural networks. In *Proceedings of the 28th International Conference on Computational Linguistics*. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Bill Dolan, Chris Quirk, and Chris Brockett. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 350–356, Geneva, Switzerland. COLING.

Luana Fagarasan, Eva Maria Vecchi, and Stephen Clark. 2015. From distributional semantics to feature norms: grounding semantic models in human perceptual data. In *Proceedings of the 11th International Conference on Computational Semantics*, pages 52–57, London, UK. Association for Computational Linguistics.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2002. Placing search in context: The concept revisited. *ACM Trans. Inf. Syst.*, 20(1):116–131.

Kristina Gulordava, Laura Aina, and Gemma Boleda. 2018a. How to represent a word and predict it, too: Improving tied architectures for language modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2936–2941, Brussels, Belgium. Association for Computational Linguistics.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018b. Colorless green recurrent networks dream hierarchically. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1195–1205, New Orleans, Louisiana. Association for Computational Linguistics.

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. SimLex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4):665–695.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, page 168–177, New York, NY, USA. Association for Computing Machinery.

Hakan Inan, Khashayar Khosravi, and Richard Socher. 2016. Tying word vectors and word classifiers: A loss framework for language modeling. *arXiv preprint arXiv:1611.01462*.

Rafal Jozefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu. 2016. Exploring the limits of language modeling. *arXiv preprint arXiv:1602.02410*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A lite BERT for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Dandan Li and Douglas Summers-Stay. 2019. Mapping distributional semantics to property norms with deep neural networks. *Big Data and Cognitive Computing*, 3(2):30.

Lucy Li and Jon Gauthier. 2017. Are distributional representations ready for the real world? Evaluating word vectors for grounded perceptual meaning. In *Proceedings of the First Workshop on Language Grounding for Robotics*, pages 76–85, Vancouver, Canada. Association for Computational Linguistics.

Tal Linzen, Grzegorz Chrupała, Yonatan Belinkov, and Dieuwke Hupkes, editors. 2019. *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Association for Computational Linguistics, Florence, Italy.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Thang Luong, Richard Socher, and Christopher Manning. 2013. Better word representations with recursive neural networks for morphology. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 104–113, Sofia, Bulgaria. Association for Computational Linguistics.

Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 216–223, Reykjavik, Iceland. European Language Resources Association (ELRA).

Rebecca Marvin and Tal Linzen. 2018. Targeted syntactic evaluation of language models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202, Brussels, Belgium. Association for Computational Linguistics.

T. Mikolov, S. Kombrink, L. Burget, J. Černocký, and S. Khudanpur. 2011. Extensions of recurrent neural network language model. In *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5528–5531.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

Andriy Mnih and Geoffrey Hinton. 2007. Three new graphical models for statistical language modelling. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, page 641–648, New York, NY, USA. Association for Computing Machinery.

Andriy Mnih and Yee Whye Teh. 2012. A fast and simple algorithm for training neural probabilistic language models. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML'12, page 419–426, Madison, WI, USA. Omnipress.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 271–278, Barcelona, Spain.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 115–124, Ann Arbor, Michigan. Association for Computational Linguistics.

Nikolaos Pappas and James Henderson. 2019. Deep residual output layers for neural language generation. volume 97 of *Proceedings of Machine Learning Research*, pages 5000–5011, Long Beach, California, USA. PMLR.

Nikolaos Pappas, Lesly Miculicich, and James Henderson. 2018. Beyond weight tying: Learning joint input-output embeddings for neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 73–83, Brussels, Belgium. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018a. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Luke Zettlemoyer, and Wen-tau Yih. 2018b. Dissecting contextual word embeddings: Architecture and representation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1499–1509, Brussels, Belgium. Association for Computational Linguistics.

Ofir Press and Lior Wolf. 2017. Using the output embedding to improve language models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 157–163, Valencia, Spain. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Kira Radinsky, Eugene Agichtein, Evgeniy Gabrilovich, and Shaul Markovitch. 2011. A word at a time: Computing word relatedness using temporal semantic analysis. In *Proceedings of the 20th International Conference on World Wide Web*, WWW '11, page 337–346, New York, NY, USA. Association for Computing Machinery.

Anna Rogers, Shashwath Hosur Ananthakrishna, and Anna Rumshisky. 2018. What's in your embedding, and how it predicts task performance. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2690–2703, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Dana Rubinstein, Effi Levi, Roy Schwartz, and Ari Rappoport. 2015. How well do distributional models capture different types of semantic knowledge? In *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)*, pages 726–730, Beijing, China. Association for Computational Linguistics.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Sho Takase, Jun Suzuki, and Masaaki Nagata. 2017. Input-to-output gate to improve RNN language models. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 43–48, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Sho Takase, Jun Suzuki, and Masaaki Nagata. 2018. Direct output connection for a high-rank language model. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4599–4609, Brussels, Belgium. Association for Computational Linguistics.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R Bowman, Dipanjan Das, et al. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. *arXiv preprint arXiv:1905.06316*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, undefinedukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Ellen M. Voorhees and Dawn M. Tice. 2000. Building a question answering test collection. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '00, page 200–207, New York, NY, USA. Association for Computing Machinery.

Eric Wallace, Yizhong Wang, Sujian Li, Sameer Singh, and Matt Gardner. 2019. Do NLP models know numbers? probing numeracy in embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5307–5315, Hong Kong, China. Association for Computational Linguistics.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39(2-3):165–210.

Haoyan Xu, Brian Murphy, and Alona Fyshe. 2016. BrainBench: A brain-image test suite for distributional semantic models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2017–2021, Austin, Texas. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Ruslan Salakhutdinov, and William W Cohen. 2017. Breaking the softmax bottleneck: A high-rank rnn language model. *arXiv preprint arXiv:1711.03953*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 5753–5763. Curran Associates, Inc.

Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. 2014. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*.