InferLite: Simple Universal Sentence Representations from Natural Language Inference Data

Jamie Ryan Kiros
Google Brain Toronto
kiros@google.com

William Chan
Google Brain Toronto
williamchan@google.com

Abstract

Natural language inference has been shown to be an effective supervised task for learning generic sentence embeddings. In order to better understand the components that lead to effective representations, we propose a lightweight version of InferSent (Conneau et al., 2017), called InferLite, that does not use any recurrent layers and operates on a collection of pre-trained word embeddings. We show that a simple instance of our model that makes no use of context, word ordering or position can still obtain competitive performance on the majority of downstream prediction tasks, with most performance gaps being filled by adding local contextual information through temporal convolutions. Our models can be trained in under 1 hour on a single GPU and allows for fast inference of new representations. Finally we describe a semantic hashing layer that allows our model to learn generic binary codes for sentences.

1 Introduction

Distributed representations of words have become immensely successful as the building blocks for deep neural networks applied to a wide range of natural language processing tasks (Pennington et al., 2014). Learning representations of sentences, however, has largely been done in a task-dependent way. In recent years, a growing body of research has emerged for learning general purpose sentence embeddings. These methods aim to learn a universal encoding function that can map arbitrary sentences into vectors which can then be applied to downstream prediction tasks without fine-tuning. Much of the motivation behind this work is to mimic the successful use of feature transfer in computer vision.

Recently, Conneau et al. (2017) showed that a bidirectional LSTM with max pooling trained to perform Natural Language Inference (NLI), called InferSent, outperforms several other encoding functions on a suite of downstream prediction tasks. This method could match or outperform existing models that learns generic embeddings in an unsupervised setting, often requiring several days or weeks to train (Kiros et al., 2015). However, a better understanding of what properties induce a useful generic embedding remains illusive.

In this work we propose a lightweight version of InferSent, called InferLite. InferLite deviates from InferSent in that it does not use any recurrent connections and can generalize to multiple pre-trained word embeddings. Our method uses a controller to dynamically weight embeddings for each word followed by max pooling over components to obtain the final sentence representation. Despite its simplicity, our method obtains performances on par with InferSent (Conneau et al., 2017) when using Glove representations (Pennington et al., 2014) as the source of pre-trained word vectors. To our surprise, the majority of evaluations can be done competitively without any notion of context, word ordering or position. For tasks where this is useful, much of the performance gap can be made up through a stack of convolutional layers to incorporate local context. Finally, we describe a semantic hashing layer that allows our model to be extended to learning generic binary vectors. The final result is a method that is both fast at training and inference and offers a strong baseline for future research on general purpose embeddings.

1.1 Why learn lightweight encoders?

Our proposed model naturally raises a question: why consider lightweight sentence encoders? If a generic encoder only needs to be trained once, why would training times be relevant? We argue our direction is important for two reasons. One is inference speed. With a lightweight encoder, we can encode millions of sentences efficiently without requiring extensive computational resources. The appendix includes inference speeds of our models. The second, perhaps more importantly, is to gain a better understanding of what properties lead to high quality generic embeddings. When models take several days or weeks to train, an ablation analysis becomes prohibitively costly. Since our models can be trained quickly, it allows for a more extensive analysis of architectural and data necessities. Moreover, we include an ablation study in the appendix that shows even innocent or seemingly irrelevant model decisions can have a drastic effect on performance. Such observations could not be observed when models take orders of magnitude longer to train.

2 Related Work

A large body of work on distributional semantics have considered encoding phrase and sentence meaning into vectors e.g. (Mitchell and Lapata, 2008; Grefenstette et al., 2013; Paperno et al., 2014). The first attempt at using neural networks for learning generic sentence embeddings was Kiros et al. (2015), who proposed a sequence-to-sequence extension of the skip-gram model but applied at the sentence level. This method was taught to encode a sentence and predict its neighbours, harnessing a large collection of books for training (Zhu et al., 2015). A similar approach, FastSent, was proposed by (Hill et al., 2016) which replaced the RNN encoder of skip-thoughts with word embedding summation. Methods using RNN encoders tend to perform poorly on STS evaluations, as shown by Wieting et al. (2015). Arora et al. (2017) showed a simple weighted bag of words with the first principal component subtracted, can be competitive on many sentencing encoding tasks.

Attempts to learn generic encoders with discriminative objectives were considered by Nie et al. (2017) and Logeswaran and Lee (2018) who replaced the decoder of skip-thoughts with classification tasks based on discourse relations and prediction of target sentences from an encoded candidate. All of the above methods relied on a large corpus of unlabelled data. Conneau et al. (2017) showed that similar or improved performance can be obtained using NLI datasets as a source of supervisory information. The state of the art sentence encoders utilize multi-task learning (Subramanian et al., 2018) by training an encoder to simultaneously do well on a collection of tasks such as NLI, next sentence prediction and translation.

The use of gating for selecting word representations has been considered in previous work. Yang et al. (2017) introduced a method for choosing between word and character embeddings while Kiela et al. (2018) describe a contextual gating method for word embedding selection. Gating has also been widely applied to multimodal fusion (Arevalo et al., 2017; Wang et al., 2018b; Kiros et al., 2018).

Feature	dataset	dim	method
Glove	Common Crawl	300	
News	Google News	500	CBOW
Query	Google Search	800	CBOW

Table 1: Comparison of word representations used.

Our work is also related to recent methods that induce contextualized word representations (McCann et al., 2017; Peters et al., 2018) as well as pre-training language models for task-dependent fine-tuning (Dai and Le, 2015; Howard and Ruder, 2018; Radford et al., 2018). We differ from these approaches in that we aim to infer a transferable sentence vector without any additional fine-tuning.

3 Method

Our method operates on a collection of pre-trained word representations and is then trained on the concatenation of SNLI (Bowman et al., 2015) and MultiNLI (Williams et al., 2018) datasets as in Conneau et al. (2017). Table 1 summarizes the properties of the embeddings we consider. At a high level, our method takes as input a collection of embeddings for each word and learns a gated controller to decide how to weight each representation. After encoding each word in a sentence, the sentence embedding is obtained by max pooling the transformed word representations. Unlike Subramanian et al. (2018), which learn a shared encoder in a multi-task setting, we instead fix the prediction task to NLI but use embeddings obtained from alternative tasks. Figure 1 illustrates our model.

We begin by defining notation. Suppose we are given a sentence of words $S = w^1, \dots, w^T$ which we would like to encode into a vector. Let K be the number of embedding types (e.g. Glove, News, Query) and let E^k denote the word embedding matrix for type k. Define $E^c = [E^1; \dots; E^K]$ to be the concatenation of word embedding matrices of all K types.

We break our model description into four modules: Encoder, Controller, Fusion and Reduction. In the appendix we include an ablation study that analyzes the effect of our design choices.

Encoder. The encoder computes $M + 1$ layers $H_0^k, \dots, H_M^k$ for $k = 1, \dots, K$ embedding types.

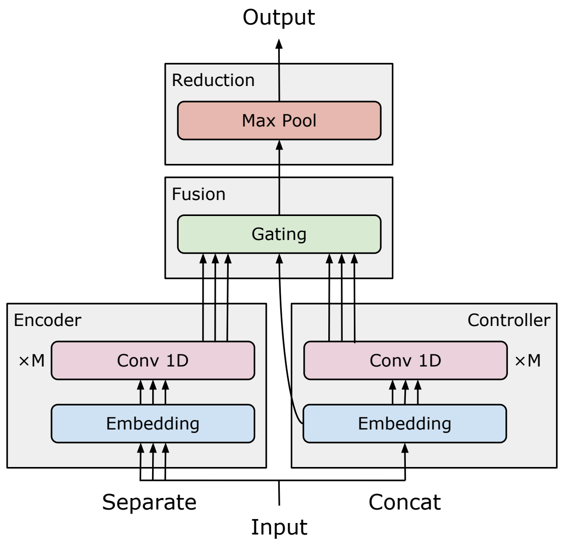


Figure 1: Illustration of the InferLite model. Separate and Concat refer to the embedding types used.

The first layer is computed as:

$$H_0^k = \phi_{0,h}(W_{0,h}^k E^k + b_{0,h}^k) \tag{1}$$

where $W_{0,g}^k E^k$ is a time distributed matrix multiply [1] and $\phi_{0,h}$ is the activation function. Each subsequent layer is given by:

$$H_i^k = \phi_{i,h}(W_{i,h}^k * H_{i-1}^k + b_{i,h}^k) \tag{2}$$

where $*$ denotes the 1-D convolution operator that preserves dimensions. Note that if the convolutional filter length is 1, the model reduces to a bag-of-words encoder. We use ReLU activation functions for $\phi_{i,h}$ where $i = 0, \ldots, M-1$ and a tanh activation for the last layer $\phi_{M,h}$.

Controller. The controller first computes a shared layer G_0^c along with M heads $G_1^k, \ldots, G_M^k$ for $k = 1, \ldots, K$ embedding types. The first layer is computed as:

$$G_0^c = \phi_{0,g}(W_{0,g}^c E^c + b_{0,g}^c) \tag{3}$$

where $W_{0,g}^c E^c$ is a time distributed matrix multiply and $\phi_{0,g}$ is the activation function. Define $G_0^k = G_0^c, k = 1, \ldots, K$. Each subsequent layer is given by:

$$G_i^k = \phi_{i,g}(W_{i,g}^k * G_{i-1}^k + b_{i,g}^k) \tag{4}$$

We use ReLU activation functions for $\phi_{i,g}$ where $i = 0, \ldots, M-1$ and a sigmoid activation for the last layer $\phi_{M,g}$.

[1]Sometimes referred to as a "translation layer" (see https://github.com/Smerity/keras_snli).

Fusion. The fusion layer combines the encoder and controller layers as follows:

$$F' = \left(\sum_{k=1}^{K} H_M^k \odot G_M^k\right) + G_0^c \tag{5}$$

$$F = \phi_f(W_f F' + b_f) \tag{6}$$

where $\odot$ denotes a component-wise product, $W_f F'$ is a time distributed matrix multiply, ϕ_f is a ReLU activation function and G_0^c is added as a skip connection. In the appendix we demonstrate that the added skip connection is crucial to the success of the model.

Reduction. The final reduction operation simply applies max pooling across tokens:

$$s = \text{maxpool}\{F\}_T \tag{7}$$

resulting in a sentence vector s. This resulting vector corresponds to the embedding for which we evaluate all downstream tasks with.

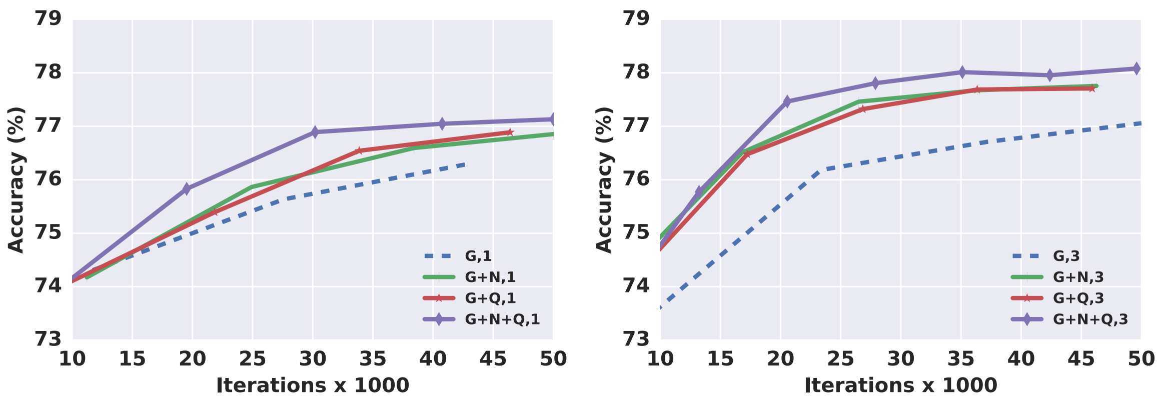
For training on NLI, we follow existing work and compute the concatenation of the embeddings of premise and hypothesis sentences along with their componentwise and absolute difference (Conneau et al., 2017). This joint vector is fed into a 2 hidden layer feedforward network with ReLU activations, followed by a softmax layer to predict whether the sentence pairs are neutral, entailed or contradictory. After training on NLI, the weights of the model are frozen and used for encoding new sentences.

3.1 Relationship to other work

Our model shares similarities to two other works, namely the gated convolutional layers from Dauphin et al. (2016); Gehring et al. (2017) and van den Oord et al. (2016). There are three main differences: 1) we generalize to multiple embedding types 2) we only apply gating at the end of the last layer as a way of weighting all embedding types (instead of each layer) and 3) we use a skip connection from the controller's transformed input to the fusion layer. We note that our encoder module can be reduced to the gated convolutional encoder in van den Oord et al. (2016) if we use one embedding type, remove the time distributed layers and only use a single convolutional layer.

3.2 Semantic hashing

We can augment a semantic hashing (Salakhutdinov and Hinton, 2009) layer to InferLite as a way of learning binary codes for sentences. Binary codes allow for efficient storage and retrieval



(a) NLI: no context (Conv length 1).

(b) NLI: with context (Conv length 3).

Figure 2: NLI accuracy using models that (a) have no context (convolution length of 1) and b) local context (convolution length of 3). Performance is reported on the concatenation of SNLI and MultiNLI development sets. **G** stands for Glove, **N** stands for News and **Q** stands for Query embeddings.

over massive corpora. To do this, we append the following layer:

$$h(s) = \sigma\left(\frac{\text{LN}(W_x s + b_x)}{\tau}\right) \qquad (8)$$

where LN is Layer Normalization (Ba et al., 2016), σ is the sigmoid activation and τ is a temperature hyperparameter. We initialize $\tau = 1$ at the beginning of training and exponentially decay τ towards 0 over the course of training. At inference time, we threshold at 0.5 to obtain codes. We found Layer Normalization was important for obtaining good codes as otherwise many dead units would form. In the appendix we include downstream performance results for 256, 1024 and 4096-bit codes. The combination of fast inference and efficient storage allows InferLite to be an effective generic encoder for large-scale retrieval and similarity search.

4 Experiments

We use the SentEval toolkit (Conneau and Kiela, 2018) for evaluating our sentence embeddings. All of our models are trained to optimize performance on the concatenation of SNLI and MultiNLI, using the concatenated development sets for early stopping. We use 4096-dimensional embeddings as in Conneau et al. (2017). We consider encoders that use convolutional filters of length 1 (no context) or length 3 (local context), with a stack of $M = 3$ convolutional layers. All word embeddings are pre-trained, normalized to unit length and held fixed during training. Full hyperparameter details are included in the appendix, including an ablation study comparing the effect of the choice of M.

We first analyze performance of our model on NLI prior to evaluating our models on downstream tasks. Figure 2 shows development set accuracy on NLI for models with and without context, using various feature combinations. Here we observe that a) using local context improves NLI performance and b) adding additional embedding types leads to improved performance.

Tables 2 and 3 show results on downstream evaluation tasks. Here several observations can be made. First note the effectiveness of the basic (glove,1) model, which is essentially a deep bag-of-unigram encoder. We also observe our models outperform all previous bag of words baselines. Next we observe that adding local context helps significantly on MR, CR, SST2 and TREC tasks. Furthermore, fusing embeddings from query and news models matches or improves performance over a glove-only model on 12 out of 15 tasks. Our (glove+news+query,3) model is best on 5 tasks and is a generally strong performer across all evaluations. Finally observe that our models significantly improves over previous work on STS tasks.

Next we compare training times of our models to previous work. All of our models can be trained in one GPU hour or less. QuickThoughts and InferSent can be trained on the order of a day while Multitask requires 1 week of training. This demonstrates the trade-off of these approaches.

In the appendix we include results from several other experiments including COCO image-sentence retrieval, downstream performance of InferLite with semantic hashing and results on 10 probing tasks introduced in Conneau et al. (2018). We also do an extensive ablation study

Model	MR	CR	SUBJ	MPQA	SST2	TREC	MRPC	time (h)
Glove BOW (Conneau et al., 2017)	78.7	78.5	91.6	87.6	79.8	83.6	72.1/80.9	0
USE-D (Cer et al., 2018)	74.5	81.0	92.7	85.4	77.6	91.2		
ST-LN (Ba et al., 2016)	79.4	83.1	93.7	89.3	82.9	88.4		∼ 720
DisSent (Nie et al., 2017)	80.1	84.9	93.6	90.1	84.1	93.6	75.0/	
InferSent (Conneau et al., 2017)	81.1	86.3	92.4	90.2	84.6	88.2	76.2/83.1	< 24
USE-T (Cer et al., 2018)	81.4	87.4	93.9	87.0	85.4	92.5		
QuickThoughts (Logeswaran and Lee, 2018)	82.4	86.0	**94.8**	90.2	**87.6**	92.4	76.9/84.0	∼ 24
Multitask (Subramanian et al., 2018)	**82.5**	**87.7**	94.0	**90.9**	83.2	**93.0**	**78.6/84.4**	∼ 168
glove,1	79.6	82.2	92.0	89.5	83.0	88.2	75.5/82.7	∼ 0.3
glove+news,1	79.0	82.7	92.1	89.8	83.7	89.2	76.9/83.7	∼ 0.5
glove+query,1	79.0	83.2	92.2	89.6	83.3	89.4	75.8/83.0	∼ 0.5
glove+news+query,1	78.8	82.2	92.0	89.6	83.0	89.2	76.7/83.5	∼ 0.7
glove,3	80.9	84.1	92.4	89.6	85.8	90.0	76.5/83.4	∼ 0.5
glove+news,3	80.4	84.8	91.9	89.7	86.3	89.8	77.0/83.9	∼ 0.7
glove+query,3	80.1	85.6	92.2	89.4	84.7	91.0	76.4/83.3	∼ 0.7
glove+news+query,3	80.4	85.6	92.2	89.9	85.0	91.2	76.1/83.3	∼ 1

Table 2: Comparison of embedding methods on downstream evaluations. Each set of results is a) bag-of-words b) RNN and Transformer c) ours, filter length 1 and d) ours, filter length 3. Last column is training time in hours.

Model	SICK-R	SICK-E	STSB	STS12	STS13	STS14	STS15	STS16
Glove BOW (Conneau et al., 2017)	80.0	78.6		52.5	42.3	54.2	52.7	
GloVe + WR (Arora et al., 2017)	86.0	84.6		56.2	56.6	68.5	71.1	
ST-LN (Ba et al., 2016)	85.8	79.5		30.8	24.8	31.4	31.0	
InferSent (Conneau et al., 2017)	88.4	86.3	75.8/75.5	59.2	58.9	69.6	71.3	71.4
Multitask (Subramanian et al., 2018)	**88.8**	**87.8**	**78.9/78.6**	60.6	54.7	65.8	74.2	66.4
glove,1	88.3	85.9	78.1/78.0	62.4	60.4	71.6	**74.6**	70.3
glove+news,1	88.5	86.7	78.0/78.1	63.0	58.8	71.2	74.2	70.2
glove+query,1	88.5	86.0	77.1/77.1	63.1	56.8	70.9	74.1	70.3
glove+news+query,1	88.6	85.9	77.7/78.0	**63.9**	58.1	70.9	73.6	69.8
glove,3	88.1	85.5	78.4/78.3	61.9	**61.3**	71.7	74.5	71.2
glove+news,3	88.5	86.6	77.5/77.4	61.7	59.5	71.0	73.9	**71.6**
glove+query,3	88.6	86.5	78.1/78.3	61.8	**61.3**	**71.8**	74.3	70.1
glove+news+query,3	88.7	87.2	77.1/77.1	63.1	58.6	71.7	73.8	70.1

Table 3: Comparison of embedding methods on downstream evaluations. Each sets of results are a) bag-of-words b) RNN encoders c) ours, filter length 1 d) ours, filter length 3. Best results bolded. Our best results underlined.

of model components and illustrate gate activation values qualitatively for sentences from the (glove+news+query,3) model.

4.1 Limitations

We also experimented with additional embedding types, including Picturebook (Kiros et al., 2018), knowledge graph and neural machine translation based embeddings. While adding these embeddings improved performance on NLI, they did not lead to any performance gains on downstream tasks. This is in contrast to Subramanian et al. (2018) who showed adding additional tasks in a multi-task objective led to better downstream performance. This demonstrates the limitations of solely using NLI as an objective, even if we transfer embeddings from additional tasks.

5 Future Work

In future work, we would like to explore using contextualized word embeddings, such as CoVe (McCann et al., 2017) and ELMo (Peters et al., 2018), as input to our models as opposed to non-contextualized representations. We also intend to evaluate on additional benchmark tasks such as GLUE (Wang et al., 2018a), explore using the learned word representations as contextualized embeddings and perform downstream fine-tuning.

Acknowledgments

The authors would like to thank Geoffrey Hinton, George Dahl and the EMNLP anonomous reviewers for their valuable feedback.

References

John Arevalo, Thamar Solorio, Manuel Montes y Gomez, and Fabio A. Gonzalez. 2017. Gated Multimodal Units for Information Fusion. In *arXiv:1702.01992*.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A Simple but Tough-to-Beat Baseline for Sentence Embeddings. In *ICLR*.

Jimmy Ba, Jamie Kiros, and Geoffrey Hinton. 2016. Layer Normalization. In *arXiv:1607.06450*.

Samuel Bowman, Gabor Angeli, Christopher Potts, and Christopher Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. 2018. Universal Sentence Encoder. In *arXiv:1803.11175*.

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. In *arXiv:1312.3005*.

Alexis Conneau and Douwe Kiela. 2018. SentEval: An Evaluation Toolkit for Universal Sentence Representations. In *arXiv:1803.05449*.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised Learning of Universal Sentence Representations from Natural Language Inference Data. In *EMNLP*.

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 *ACL*.

Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In *NIPS*.

Yann N Dauphin, Angela Fan, Michael Auli, and David Grangier. 2016. Language modeling with gated convolutional networks. In *arXiv:1612.08083*.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. In *arXiv:1705.03122*.

Edward Grefenstette, Georgiana Dinu, Yao-Zhong Zhang, Mehrnoosh Sadrzadeh, and Marco Baroni. 2013. Multi-step regression learning for compositional distributional semantics. In *arXiv:1301.6939*.

Felix Hill, Kyunghyun Cho, and Anna Korhonen. 2016. Learning distributed representations of sentences from unlabelled data. In *NAACL*.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *ACL*.

Douwe Kiela, Changhan Wang, and Kyunghyun Cho. 2018. Context-Attentive Embeddings for Improved Sentence Representations. In *arXiv:1804.07983*.

Diederik Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*.

Jamie Kiros, William Chan, and Geoffrey Hinton. 2018. Illustrative Language Understanding: Large-Scale Visual Grounding with Image Search. In *ACL*.

Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. 2015. Skip-Thought Vectors. In *NIPS*.

Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, Lawrence Zitnick, and Piotr Dollr. 2014. Microsoft COCO: Common Objects in Context. In *ECCV*.

Lajanugen Logeswaran and Honglak Lee. 2018. An efficient framework for learning sentence representations. In *ICLR*.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. In *NIPS*.

Jeff Mitchell and Mirella Lapata. 2008. Vector-based models of semantic composition. In *ACL: HLT*.

Allen Nie, Erin D Bennett, and Noah D Goodman. 2017. Dissent: Sentence representation learning from explicit discourse relations. In *arXiv:1710.04334*.

Aaron van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. 2016. Conditional image generation with pixelcnn decoders. In *NIPS*.

Denis Paperno, Marco Baroni, et al. 2014. A practical and linguistically-motivated approach to compositional distributional semantics. In *ACL*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *EMNLP*.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL*.

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

Ruslan Salakhutdinov and Geoffrey Hinton. 2009. Semantic hashing. In *International Journal of Approximate Reasoning*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. In *JMLR*.

Sandeep Subramanian, Adam Trischler, Yoshua Bengio, and Christopher Pal. 2018. Learning general purpose distributed sentence representations via large scale multi-task learning. In *ICLR*.

Alex Wang, Amapreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018a. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *arXiv:1804.07461*.

Shaonan Wang, Jiajun Zhang, and Chengqing Zong. 2018b. Learning Multimodal Word Representation via Dynamic Fusion Methods. In *AAAI*.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2015. Towards universal paraphrastic sentence embeddings. In *arXiv:1511.08198*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. In *NAACL*.

Zhilin Yang, Bhuwan Dhingra, Ye Yuan, Junjie Hu, William W Cohen, and Ruslan Salakhutdinov. 2017. Words or Characters? Fine-grained Gating for Reading Comprehension. In *ICLR*.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *ICCV*.