Universal Sentence Encoder

Daniel Cer\textsuperscript{a}, Yinfei Yang\textsuperscript{a}, Sheng-yi Kong\textsuperscript{a}, Nan Hua\textsuperscript{a}, Nicole Limtiaco\textsuperscript{b}, Rhomni St. John\textsuperscript{a}, Noah Constant\textsuperscript{a}, Mario Guajardo-Céspedes\textsuperscript{a}, Steve Yuan\textsuperscript{c}, Chris Tar\textsuperscript{a}, Yun-Hsuan Sung\textsuperscript{a}, Brian Strope\textsuperscript{a}, Ray Kurzweil\textsuperscript{a}

\textsuperscript{a}Google Research
Mountain View, CA  
\textsuperscript{b}Google Research
New York, NY  
\textsuperscript{c}Google
Cambridge, MA

Abstract

We present models for encoding sentences into embedding vectors that specifically target transfer learning to other NLP tasks. The models are efficient and result in accurate performance on diverse transfer tasks. Two variants of the encoding models allow for trade-offs between accuracy and compute resources. For both variants, we investigate and report the relationship between model complexity, resource consumption, the availability of transfer task training data, and task performance. Comparisons are made with baselines that use word level transfer learning via pretrained word embeddings as well as baselines do not use any transfer learning. We find that transfer learning using sentence embeddings tends to outperform word level transfer. With transfer learning via sentence embeddings, we observe surprisingly good performance with minimal amounts of supervised training data for a transfer task. We obtain encouraging results on Word Embedding Association Tests (WEAT) targeted at detecting model bias. Our pre-trained sentence encoding models are made freely available for download and on TF Hub.

1 Introduction

Limited amounts of training data are available for many NLP tasks. This presents a challenge for data hungry deep learning methods. Given the high cost of annotating supervised training data, very large training sets are usually not available for most research or industry NLP tasks. Many models address the problem by implicitly performing limited transfer learning through the use of pre-trained word embeddings such as those produced by word2vec (Mikolov et al., 2013) or GloVe (Pennington et al., 2014). However, recent work has demonstrated strong transfer task performance using pre-trained sentence level embeddings (Conneau et al., 2017).

In this paper, we present two models for producing sentence embeddings that demonstrate good transfer to a number of other of other NLP tasks. We include experiments with varying amounts of transfer task training data to illustrate the relationship between transfer task performance and training set size. We find that our sentence embeddings can be used to obtain surprisingly good task performance with remarkably little task specific training data. The sentence encoding models are made publicly available on TF Hub.

Engineering characteristics of models used for transfer learning are an important consideration. We discuss modeling trade-offs regarding memory requirements as well as compute time on CPU and GPU. Resource consumption comparisons are made for sentences of varying lengths.
import tensorflow_hub as hub
embed = hub.Module("https://tfhub.dev/google/
   "universal-sentence-encoder/1")
embedding = embed(
   "The quick brown fox jumps over the lazy dog.")

Listing 1: Python example code for using the universal sentence encoder.

2 Model Toolkit

We make available two new models for encoding sentences into embedding vectors. One makes use of the transformer (Vaswani et al., 2017) architecture, while the other is formulated as a deep averaging network (DAN) (Iyyer et al., 2015). Both models are implemented in TensorFlow (Abadi et al., 2016) and are available to download from TF Hub:

https://tfhub.dev/google/
universal-sentence-encoder/1

The models take as input English strings and produce as output a fixed dimensional embedding representation of the string. Listing 1 provides a minimal code snippet to convert a sentence into a tensor containing its sentence embedding. The embedding tensor can be used directly or incorporated into larger model graphs for specific tasks.

As illustrated in Figure 1, the sentence embeddings can be trivially used to compute sentence level semantic similarity scores that achieve excellent performance on the semantic textual similarity (STS) Benchmark (Cer et al., 2017). When included within larger models, the sentence encoding models can be fine tuned for specific tasks using gradient based updates.

3 Encoders

We introduce the model architecture for our two encoding models in this section. Our two encoders have different design goals. One based on the transformer architecture targets high accuracy at the cost of greater model complexity and resource consumption. The other targets efficient inference with slightly reduced accuracy.

3.1 Transformer

The transformer based sentence encoding model constructs sentence embeddings using the encoding sub-graph of the transformer architecture (Vaswani et al., 2017). This sub-graph uses attention to compute context aware representations of words in a sentence that take into account both the ordering and identity of all the other words. The context aware word representations are converted to a fixed length sentence encoding vector by computing the element-wise sum of the representations at each word position.\(^1\) The encoder takes as input a lowercased PTB tokenized string and outputs a 512 dimensional vector as the sentence embedding.

The encoding model is designed to be as general purpose as possible. This is accomplished by using multi-task learning whereby a single encoding model is used to feed multiple downstream tasks. The supported tasks include: a Skip-Thought like task (Kiros et al., 2015) for the unsupervised learning from arbitrary running text; a conversational input-response task for the inclusion of parsed conversational data (Henderson et al., 2017); and classification tasks for training on supervised data. The Skip-Thought task replaces the LSTM (Hochreiter and Schmidhuber, 1997) used in the original formulation with a model based on the Transformer architecture.

As will be shown in the experimental results below, the transformer based encoder achieves the best overall transfer task performance. However, this comes at the cost of compute time and memory usage scaling dramatically with sentence length.

3.2 Deep Averaging Network (DAN)

The second encoding model makes use of a deep averaging network (DAN) (Iyyer et al., 2015) whereby input embeddings for words and bi-grams are first averaged together and then passed through a feedforward deep neural network (DNN) to produce sentence embeddings. Similar to the Transformer encoder, the DAN encoder takes as input a lowercased PTB tokenized string and outputs a 512 dimensional sentence embedding. The DAN encoder is trained similarly to the Transformer based encoder. We make use of mul-

\(^{1}\)The encoding model for the DAN based encoder is already available. The transformer based encoder will be made available at a later point.

\(^{2}\)Visit https://colab.research.google.com/ to try the code snippet in Listing 1. Example code and documentation is available on the universal encoder website provided above.
titask learning whereby a single DAN encoder is used to supply sentence embeddings for multiple downstream tasks.

The primary advantage of the DAN encoder is that compute time is linear in the length of the input sequence. Similar to Iyyer et al. (2015), our results demonstrate that DANs achieve strong baseline performance on text classification tasks.

### 3.3 Encoder Training Data

Unsupervised training data for the sentence encoding models are drawn from a variety of web sources. The sources are Wikipedia, web news, web question-answer pages and discussion forums. We augment unsupervised learning with training on supervised data from the Stanford Natural Language Inference (SNLI) corpus (Bowman et al., 2015). Similar to the findings of Conneau et al. (2017), we observe that training to SNLI improves transfer performance.

### 4 Transfer Tasks

This section presents an overview of the data used for the transfer learning experiments and the Word Embedding Association Test (WEAT) data used to characterize model bias. Table 1 summarizes the number of samples provided by the test portion of each evaluation set and, when available, the size of the dev and training data.

| Dataset      | Train | Dev | Test  |
|--------------|-------|-----|-------|
| SST          | 67,349| 872 | 1,821 |
| STS Bench    | 5,749 | 1,500| 1,379 |
| TREC         | 5,452 | -   | 500   |
| MR           | -     | -   | 10,662|
| CR           | -     | -   | 3,775 |
| SUBJ         | -     | -   | 10,000|
| MPQA         | -     | -   | 10,606|

Table 1: Transfer task evaluation sets

### 5 Transfer Learning Models

For sentence classification transfer tasks, the output of the transformer and DAN sentence encoders are provided to a task specific DNN. For the pairwise semantic similarity task, we directly assess the similarity of the sentence embeddings produced by our two encoders. As shown Eq. 1, we first compute the cosine similarity of the two sentence embeddings and then use arccos to convert the cosine similarity into an angular distance.

\[
sim(u, v) = \left(1 - \arccos\left(\frac{u \cdot v}{||u|| ||v||}\right)/\pi\right) (1)
\]

#### 5.1 Baselines

For each transfer task, we include baselines that only make use of word level transfer and baselines that make use of no transfer learning at all. For word level transfer, we use word embeddings from a word2vec skip-gram model trained on a corpus of news data (Mikolov et al., 2013). The pretrained word embeddings are included as input to two model types: a convolutional neural network models (CNN) (Kim, 2014); a DAN. The baselines that use pretrained word embeddings allow us to contrast word versus sentence level transfer. Additional baseline CNN and DAN models are trained without using any pretrained word or sentence embeddings.

#### 5.2 Combined Transfer Models

We explore combining the sentence and word level transfer models by concatenating their representations prior to feeding the combined representation

---

4For the datasets MR, CR, and SUBJ, SST, and TREC we use the preparation of the data provided by Conneau et al. (2017).

5We find that using a similarity based on angular distance performs better on average than raw cosine similarity.
Table 2: Model performance on transfer tasks. *USE_T* is the universal sentence encoder (USE) using Transformer. *USE_D* is the universal encoder DAN model. Models tagged with *w2v w.e.* make use of pre-training word2vec skip-gram embeddings for the transfer task model, while models tagged with *lrn w.e.* use randomly initialized word embeddings that are learned only on the transfer task data. Accuracy is reported for all evaluations except STS Bench where we report the Pearson correlation of the similarity scores with human judgments. Pairwise similarity scores are computed directly using the sentence embeddings from the universal sentence encoder as in Eq. (1).

| Model                  | MR  | CR  | SUBJ | MPQA | TREC | SST  | STS Bench  |
|------------------------|-----|-----|------|------|------|------|------------|
|                         | Sent & Word Embedding | Transfer Learning |        |       |      |      | (dev / test) |
| USE_D+DAN (w2v w.e.)   | 77.11 | 81.71 | 93.12 | 87.01 | 94.72 | 82.14 | –          |
| USE_D+CNN (w2v w.e.)   | 78.20 | 82.04 | 93.24 | 85.87 | 97.67 | 85.29 | –          |
| USE_T+DAN (w2v w.e.)   | 81.32 | 86.66 | 93.90 | 88.14 | 95.51 | 86.62 | –          |
| USE_T+CNN (w2v w.e.)   | 81.18 | 87.45 | 93.58 | 87.32 | 98.07 | 86.69 | –          |

**Sentence Embedding Transfer Learning**

| USE_D        | 74.45 | 80.97 | 92.65 | 85.38 | 91.19 | 77.62 | 0.763 / 0.719 (r) |
| USE_T        | 81.44 | 87.43 | 93.87 | 86.98 | 92.51 | 85.38 | 0.814 / 0.782 (r) |
| USE_D+CNN (lrn w.e.) | 77.57 | 81.93 | 92.91 | 85.97 | 95.86 | 83.41 | –          |
| USE_T+CNN (lrn w.e.) | 78.49 | 81.49 | 92.99 | 85.53 | 97.71 | 85.27 | –          |
| USE_T+DAN (lrn w.e.) | 81.36 | 86.08 | 93.66 | 87.14 | 96.60 | 86.24 | –          |
| USE_T+CNN (lrn w.e.) | 81.59 | 86.45 | 93.36 | 86.85 | 97.44 | 87.21 | –          |

**Word Embedding Transfer Learning**

| DAN (w2v w.e.) | 74.75 | 75.24 | 90.80 | 81.25 | 85.69 | 80.24 | –          |
| CNN (w2v w.e.) | 75.10 | 80.18 | 90.84 | 81.38 | 97.32 | 83.74 | –          |

**Baselines with No Transfer Learning**

| DAN (lrn w.e.) | 75.97 | 76.91 | 89.49 | 80.93 | 93.88 | 81.52 | –          |
| CNN (lrn w.e.) | 76.39 | 79.39 | 91.18 | 82.20 | 95.82 | 84.90 | –          |

Table 2: Model performance on transfer tasks. *USE_T* is the universal sentence encoder (USE) using Transformer. *USE_D* is the universal encoder DAN model. Models tagged with *w2v w.e.* make use of pre-training word2vec skip-gram embeddings for the transfer task model, while models tagged with *lrn w.e.* use randomly initialized word embeddings that are learned only on the transfer task data. Accuracy is reported for all evaluations except STS Bench where we report the Pearson correlation of the similarity scores with human judgments. Pairwise similarity scores are computed directly using the sentence embeddings from the universal sentence encoder as in Eq. (1).

To assess bias in our encoding models, we evaluate the strength of various associations learned by our model on WEAT word lists. We compare our result to those of Caliskan et al. (2017) who discovered that word embeddings could be used to reproduce human performance on implicit association tasks for both benign and potentially undesirable associations.

### 6 Experiments

Transfer task model hyperparameters are tuned using a combination of Vizier (Golovin et al.) and light manual tuning. When available, model hyperparameters are tuned using task dev sets. Otherwise, hyperparameters are tuned by cross-validation on the task training data when available or the evaluation test data when neither training nor dev data are provided. Training repeats ten times for each transfer task model with different randomly initialized weights and we report evaluation results by averaging across runs.

Transfer learning is critically important when training data for a target task is limited. We explore the impact on task performance of varying the amount of training data available for the task both with and without the use of transfer learning. Contrasting the transformer and DAN based encoders, we demonstrate trade-offs in model complexity and the amount of data required to reach a desired level of accuracy on a task.

Table 3 illustrates transfer task performance for varying amounts of training data. We observe that, for smaller quantities of data, sentence level transfer learning can achieve surprisingly good task performance.
Table 3: Task performance on SST for varying amounts of training data. SST 67.3k represents the full training set. Using only 1,000 examples for training, transfer learning from USE_T is able to obtain performance that rivals many of the other models trained on the full 67.3 thousand example training set.

performance. As the training set size increases, models that do not make use of transfer learning approach the performance of the other models.

Table 4 contrasts Caliskan et al. (2017)’s findings on bias within GloVe embeddings with the DAN variant of the universal encoder. Similar to GloVe, our model reproduces human associations between flowers vs. insects and pleasantness vs. unpleasantness. However, our model demonstrates weaker associations than GloVe for probes targeted at revealing ageism, racism and sexism. The differences in word association patterns can be attributed to differences in the training data composition and the mixture of tasks used to train the sentence embeddings.

7.1 Discussion

Transfer learning leads to performance improvements on many tasks. Using transfer learning is more critical when less training data is available. When task performance is close, the correct modeling choice should take into account engineering trade-offs regarding the memory and compute resource requirements introduced by the different models that could be used.

8 Resource Usage

This section describes memory and compute resource usage for the transformer and DAN sentence encoding models for different sentence lengths. Figure 2 plots model resource usage against sentence length.

**Compute Usage** The transformer model time complexity is $O(n^2)$ in sentence length, while the DAN model is $O(n)$. As seen in Figure 2 (a-b), for short sentences, the transformer encoding model is only moderately slower than the much simpler DAN model. However, compute time for transformer increases noticeably as sentence length increases. In contrast, the compute time for the DAN model stays nearly constant as sentence length is increased. Since the DAN model is remarkably computational efficient, using GPUs over CPUs will often have a much larger practical impact for the transformer based encoder.

**Memory Usage** The transformer model space complexity also scales quadratically, $O(n^2)$, in sentence length, while the DAN model space complexity is constant in the length of the sentence.
(a) CPU Time vs. Sentence Length  
(b) GPU Time vs. Sentence Length  
(c) Memory vs. Sentence Length

Figure 2: Model Resource Usage for both USE_D and USE_T at different batch sizes and sentence lengths.

| Target words                          | Attrib. words                        | Ref | GloVe  | Uni. Enc. (DAN) |
|---------------------------------------|--------------------------------------|-----|--------|-----------------|
|                                       |                                      |     |        |                 |
| Eur.-American vs. Afr.-American names | Pleasant vs. Unpleasant              | a   | 1.41   | 0.361           |
| Eur.-American vs. Afr.-American names | Pleasant vs. Unpleasant from (a)     | b   | 1.50   | -0.372          |
| Eur.-American vs. Afr.-American names | Pleasant vs. Unpleasant from (c)     | b   | 1.28   | 0.721           |
| Male vs. female names                 | Career vs. family                    | c   | 1.81   | 0.0248          |
| Math vs. arts                         | Male vs. female terms                | c   | 1.06   | 0.588           |
| Science vs. arts                      | Male vs. female terms                | d   | 1.24   | 0.236           |
| Mental vs. physical disease           | Temporary vs. permanent              | e   | 1.38   | 1.01            |
| Young vs. old peoples names           | Pleasant vs. unhappy                 | c   | 1.21   | 0.0027          |
| Flowers vs. insects                   | Pleasant vs. Unpleasant              | a   | 1.50   | 1.38            |
| Instruments vs. Weapons               | Pleasant vs. Unpleasant              | a   | 1.53   | 1.44            |

Table 4: Word Embedding Association Tests (WEAT) for GloVe and the Universal Encoder. Effect size is reported as Cohen’s d over the mean cosine similarity scores across grouped attribute words. Statistical significance is reported for 1 tailed p-scores. The letters in the Ref column indicates the source of the IAT word lists: (a) Greenwald et al. (1998) (b) Bertrand and Mullainathan (2004) (c) Nosek et al. (2002a) (d) Nosek et al. (2002b) (e) Monteith and Pettit (2011).

Similar to compute usage, memory usage for the transformer model increases quickly with sentence length, while the memory usage for the DAN model remains constant. We note that, for the DAN model, memory usage is dominated by the parameters used to store the model unigram and bigram embeddings. Since the transformer model only needs to store unigram embeddings, for short sequences it requires nearly half as much memory as the DAN model.

9 Conclusion

Both the transformer and DAN based universal encoding models provide sentence level embeddings that demonstrate strong transfer performance on a number of NLP tasks. The sentence level embeddings surpass the performance of transfer learning using word level embeddings alone. Models that make use of sentence and word level transfer achieve the best overall performance. We observe that transfer learning is most helpful when limited training data is available for the transfer task. The encoding models make different trade-offs regarding accuracy and model complexity that should be considered when choosing the best model for a particular application. The pre-trained encoding models will be made publicly available for research and use in applications that can benefit from a better understanding of natural language.

Acknowledgments

We thank our teammates from Descartes, Ai.h and other Google groups for their feedback and suggestions. Special thanks goes to Ben Packer and Yoni Halpern for implementing the WEAT assessments and discussions on model bias.

References

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard,
