NO TRAINING REQUIRED: EXPLORING RANDOM ENCODERS FOR SENTENCE CLASSIFICATION

John Wieting & Douwe Kiela

Related works

Methods include autoencoders(Socher et al., 2011; Hill et al., 2016)

Methods include other learning frameworks using raw text (Le & Mikolov, 2014; Pham et al., 2015; Jernite et al., 2017; Pagliardini et al., 2017)

Methods include a collection of books (Kiros et al., 2015)

Methods include labelled entailment corpora (Conneau et al., 2017)

Methods include image-caption data (Kiela et al., 2017)

Methods include raw text labelled with discourse relations (Nie et al., 2017)

Methods include parallel corpora (Wieting & Gimpel, 2017)

Multi-task combinations of these approaches (Subramanian et al., 2018; Cer et al., 2018)

Approach

In this paper, three architectures that produce sentence embeddings from pre-trained word embeddings, without requiring any training of the encoder itself were explored. These sentence embeddings are then used as features for a collection of downstream tasks. The downstream tasks are all trained with a logistic regression classifier using the default settings of the SentEval framework. The parameters of this classifier are the only ones that are updated during training.

RANDOM SENTENCE ENCODERS

They are concerned with obtaining a good sentence representation h that is computed using some function f parameterized by θ over pre-trained input word embeddings e ∈ L, i.e. h = fθ(e1, . . . , en) where ei is the embedding for the i-th word in a sentence of length n. Typically, sentence encoders learn θ, after which it is kept fixed for the transfer tasks. InferSent represents a sentence as f = max(BiLSTM(e1, . . . , en)) and optimizes the parameters using a supervised cross-entropy objective for predicting one of three labels from a combination of two sentence representations: entailment, neutral or contradictive. SkipThought represents a sentence as f = GRUn(e1, . . . , en), with the objective of being able to decode the previous and next utterance using negative log-likelihood from the final (i.e., n-th) hidden state. InferSent requires large amounts of expensive annotation, while SkipThought takes a very long time to train. They experiment with three methods for computing h: Bag of random embedding projections, Random LSTMs, and Echo State Networks.

BAG OF RANDOM EMBEDDING PROJECTIONS (BOREP)

The first family of architectures we explore consists of simply applying a single random projection in a standard bag-of-words (or more accurately, bag-of-embeddings) model. We randomly initialize a matrix $W \in \mathbb{R}^{D \times d}$, where D is the dimension of the projection and d is the dimension of our input word embedding. The values for the matrix are sampled uniformly from $[-\frac{1}{\sqrt{d}}, \frac{1}{\sqrt{d}}]$, which is a standard initialization heuristic used in neural networks (Glorot & Bengio, 2010). The sentence representation is then obtained as follows:

$$\mathbf{h} = f_{pool}(W\mathbf{e}_i),$$

where f_{pool} is some pooling function, e.g. $f_{pool}(x) = \sum(x)$, $f_{pool}(x) = \max(x)$ (max pooling) or $f_{pool}(x) = |x|^{-1} \sum(x)$ (mean pooling). Optionally, we impose a nonlinearity $max(0, \mathbf{h})$. We experimented with imposing positional encoding for the word embeddings, but did not find this to help.

RANDOM LSTMS

Following InferSent, we employ bidirectional LSTMs, but in our case without any training. Conneau et al. (2017) reported good performance for the random LSTM model on the transfer tasks. The LSTM weight matrices and their corresponding biases are initialized uniformly at random from $[-\frac{1}{\sqrt{d}}, \frac{1}{\sqrt{d}}]$, where d is the hidden size of the LSTM. In other words, the architecture here is the same as that of InferSent modulo the type of pooling used:

$$\mathbf{h} = f_{pool}(\text{BiLSTM}(\mathbf{e}_1, \ldots, \mathbf{e}_n)).$$

ECHO STATE NETWORKS

Echo State Networks (ESNs) (Jaeger, 2001) were primarily designed for sequence prediction problems, where given a sequence X, we predict a label y for each step in the sequence. The goal is to minimize the error between the predicted $\hat{y}$ and the target y at each timestep. Formally, an ESN is described using the following update equations:

$$\tilde{\mathbf{h}}_i = f_{pool}(W^i \mathbf{e}_i + W^h \mathbf{h}_{i-1} + b^i)$$

$$\mathbf{h}_i = (1 - \alpha)\mathbf{h}_{i-1} + \alpha \tilde{\mathbf{h}}_i,$$

Performance on all ten downstream tasks where all models have 4096 dimensions with the exception of BOE (300) and ST-LN (4800)

Model	Dim	MR	CR	MPQA	SUBJ	SST2	TREC	SICK-R	SICK-E	MRPC	STSB
BOE	300	77.3(.2)	78.6(.3)	87.6(.1)	91.3(.1)	80.0(.5)	81.5(.8)	80.2(.1)	78.7(.1)	72.9(.3)	70.5(.1)
BOREP	4096	77.4(.4)	79.5(.2)	88.3(.2)	91.9(.2)	81.8(.4)	**88.8(.3)**	85.5(.1)	82.7(.7)	73.9(.4)	68.5(.6)
RandLSTM	4096	77.2(.3)	78.7(.5)	87.9(.1)	91.9(.2)	81.5(.3)	86.5(1.1)	85.5(.1)	81.8(.5)	**74.1(.5)**	72.4(.5)
ESN	4096	**78.1(.3)**	**80.0(.6)**	**88.5(.2)**	**92.6(.1)**	**83.0(.5)**	87.9(1.0)	**86.1(.1)**	**83.1(.4)**	73.4(.4)	**74.4(.3)**
InferSent-1 = paper, G	4096	81.1	86.3	90.2	92.4	84.6	88.2	88.3	86.3	76.2	75.6
InferSent-2 = fixed pad, F	4096	79.7	84.2	89.4	92.7	84.3	90.8	88.8	86.3	76.0	78.4
InferSent-3 = fixed pad, G	4096	79.7	83.4	88.9	92.6	83.5	90.8	88.5	84.1	76.4	77.3
Δ InferSent-3, BestRand	-	*1.6*	*3.4*	*0.4*	*0.0*	*0.5*	*2.0*	*2.4*	*1.0*	*2.3*	*2.9*
ST-LN	4800	79.4	83.1	89.3	93.7	82.9	88.4	85.8	79.5	73.2	68.9
Δ ST-LN, BestRand	-	*1.3*	*3.1*	*0.8*	*1.1*	*-0.1*	*0.5*	*-0.3*	*-3.6*	*-0.9*	*-5.5*

EVALUATION

In our experiments, we evaluate on a standard sentence representation benchmark using SentEval (Conneau & Kiela, 2018). SentEval allows for evaluation on both downstream NLP datasets as well as probing tasks, which measure how accurately a representation can predict linguistic information about a given sentence. The set of downstream tasks we use for evaluation comprises sentiment analysis (MR, SST), question-type (TREC), product reviews (CR), subjectivity (SUBJ), opinion polarity (MPQA), paraphrasing (MRPC), entailment (SICK-E, SNLI) and semantic relatedness (SICK-R, STSB). The probing tasks consist of those in Conneau et al. (2018). We use the default SentEval settings (see Appendix A).

RESULTS

They compare primarily to two well-studied sentence embedding models, InferSent (Conneau et al., 2017) and SkipThought (Kiros et al., 2015) with layer normalization (Ba et al., 2016). There are recently introduced multi-task sentence encoders that improve performance further, but these either do not use pre-trained word embeddings. They compute the average accuracy/Pearson's r, along with the standard deviation, over 5 different seeds for the random methods, and tune on validation for each task.

TAKING COVER TO THE MAX

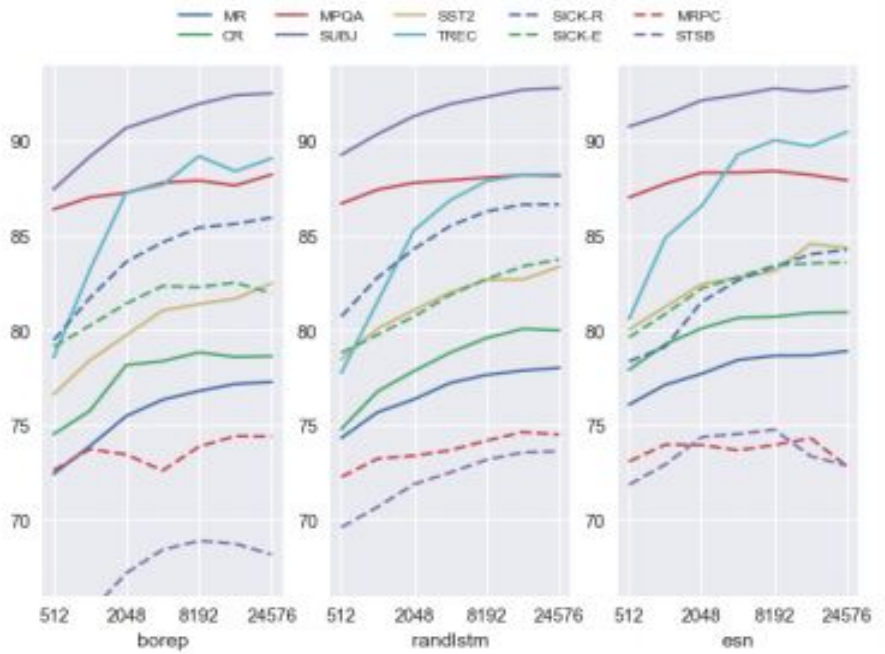
If we take Cover's theorem to the limit, we can project to an even higher-dimensional representation as long as we can still easily fit things onto a modern GPU: hence, we project to 4096 × 6 (24576) dimensions instead of the 4096 dimensions we used in previous table. In order to make for a fair comparison, we can also randomly project InferSent and SkipThought representations to the same dimensionality and examine performance.

Performance on all ten downstream tasks

Model	MR	CR	MPQA	SUBJ	SST2	TREC	SICK-R	SICK-E	MRPC	STSB
BOE	77.3(.2)	78.6(.3)	87.6(.1)	91.3(.1)	80.0(.5)	81.5(.8)	80.2(.1)	78.7(.1)	72.9(.3)	70.5(.1)
BOREP	78.6(.2)	79.9(.4)	88.8(.1)	93.0(.1)	82.5(.8)	89.5(1.3)	85.9(.0)	84.3(.3)	73.7(.9)	68.3(.5)
RandLSTM	78.2(.2)	79.9(.4)	88.2(.2)	92.8(.2)	83.2(.4)	88.4(.7)	86.6(.1)	83.0(.9)	74.7(.4)	**73.6(.4)**
ESN	**79.1(.2)**	**80.2(.3)**	**88.9(.1)**	**93.4(.2)**	**84.6(.5)**	**92.2(.8)**	**87.2(.1)**	**85.1(.2)**	**75.3(.6)**	73.1(.2)
InferSent-3 4096×6	**79.7**	**83.9**	**89.1**	**92.8**	**82.4**	**90.6**	79.5	**85.9**	**75.1**	**75.0**
ST-LN 4096×6	75.2	80.8	86.8	92.7	80.6	88.4	**82.9**	81.3	71.5	67.0

Standard deviations are show in parentheses. All models have 4096×6 dimensions. ST-LN and InferSent-3 were projected to this dimension with a random projection. Interestingly, the gap seems to get smaller, and the projection in fact appears to be detrimental to InferSent and SkipThought performance. The numbers reported in the table are competitive with (older) much more sophisticated trained methods.

Performance while varying dimensionality, for the three random sentence encoders over all ten downstream tasks.



ANALYSIS

They analyze random sentence embeddings by examining how these embeddings perform on the probing tasks introduced by Conneau et al. (2018), in order to gauge what properties of sentences they are able to recover. These probing tasks were introduced in order to provide a framework for ascertaining the linguistic properties of sentence embeddings, comprising three types of information: surface, syntactic and semantic information.

Performance on a set of probing tasks defined in

Model	SentLen	WC	TreeDepth	TopConst	BShift	Tense	SubjNum	ObjNum	SOMO	CoordInv
BOE (300d, class.)	60.5	87.5	32.0	62.7	50.0	83.7	78.0	76.6	50.5	53.8
BOREP (4096d, class.)	64.4	**97.1**	33.0	71.3	49.8	86.3	81.5	79.3	49.5	54.1
RandLSTM (4096d, class.)	72.8	94.1	35.6	76.2	55.2	86.6	84.0	79.5	49.7	63.1
ESN (4096d, class.)	78.8	92.4	36.9	76.2	62.9	86.6	82.3	79.7	49.7	60.3
Infersent-3	**80.6**	93.5	37.1	78.2	57.3	86.8	84.8	80.5	53.0	65.8
ST-LN	79.9	79.9	**39.5**	**82.1**	**69.4**	**90.2**	**86.2**	**83.4**	**54.5**	**68.9**

This table shows the performance of the random sentence encoders (using the best-overall model tuned on the classification validation sets of the SentEval tasks) on these probing tasks along with bag-ofembeddings (BOE), SkipThought-LN, and InferSent. From the table, we see that ESNs and RandLSTMs outperform BOE and BOREP on most of the tasks, especially those that require knowledge of the order of the words. This indicates that these models, even though initialized randomly, are capturing order, as one would expect. We also see that ESNs and InferSent are fairly close on many of the tasks, with Skipthought-LN generally outperforming both.

DISCUSSION

List several take-away messages with regard to sentence embeddings:

- If you need a baseline for your sentence encoder, don't just use BOE, use BOREP of the same dimension, and/or a randomly initialized version of your encoder.
- If you are pressed for time and have a small to mid-size dataset, simply randomly project to a very high dimensionality, and profit.
- More dimensions in the encoder is usually better (up to a point).
- If you want to show that your system is better than another system, use the same classifier on top with the same hyperparameters; and use the same word embeddings at the bottom; while having the same sentence embedding dimensionality.
- Be careful with padding, pooling and sorting: you may inadvertently end up favoring certain methods on some tasks, making it harder to identify sources of improvement.

CONCLUSION

In this work they have sought to put sentence embeddings on more solid footing by examining how much trained sentence encoders improve over random sentence encoders. As it turns out, differences exist, but are smaller than they would have hoped: in comparison to sentence encoders such as SkipThought (which was trained for a very long time) and InferSent (which requires large amounts of annotated data), performance improvements are less than 2 points on average over the 10 SentEval tasks. Therefore one may wonder to what extent sentence encoders are worth the attention they're receiving. Hope remains, however, if they as a community start focusing on more sophisticated tasks that require more sophisticated learned representations that cannot merely rely on having good pre-trained word embeddings.