Gating Mechanisms for Combining Character and Word-level Word Representations: An Empirical Study

Jorge A. Balazs and **Yutaka Matsuo**
Graduate School of Engineering
The University of Tokyo
{jorge, matsuo}@weblab.t.u-tokyo.ac.jp

Abstract

In this paper we study how different ways of combining character and word-level representations affect the quality of both final word and sentence representations. We provide strong empirical evidence that modeling characters improves the learned representations at the word and sentence levels, and that doing so is particularly useful when representing less frequent words. We further show that a feature-wise sigmoid gating mechanism is a robust method for creating representations that encode semantic similarity, as it performed reasonably well in several word similarity datasets. Finally, our findings suggest that properly capturing semantic similarity at the word level does not consistently yield improved performance in downstream sentence-level tasks. Our code is available at `https://github.com/jabalazs/gating`.

1 Introduction

Incorporating sub-word structures like substrings, morphemes and characters to the creation of word representations significantly increases their quality as reflected both by intrinsic metrics and performance in a wide range of downstream tasks (Bojanowski et al., 2017; Luong and Manning, 2016; Wu et al., 2016; Ling et al., 2015).

The reason for this improvement is related to sub-word structures containing information that is usually ignored by standard word-level models. Indeed, when representing words as vectors extracted from a lookup table, semantically related words resulting from inflectional processes such as *surf*, *surfing*, and *surfed*, are treated as being independent from one another[1]. Further, word-level embeddings do not account for derivational processes resulting in syntactically-similar words with different meanings such as *break*, *breakable*, and *unbreakable*. This causes derived words, which are usually less frequent, to have lower-quality (or no) vector representations.

Previous works have successfully combined character-level and word-level word representations, obtaining overall better results than using only word-level representations. For example Luong and Manning (2016) achieved state-of-the-art results in a machine translation task by representing unknown words as a composition of their characters. Botha and Blunsom (2014) created word representations by adding the vector representations of the words' surface forms and their morphemes ($\overrightarrow{\text{perfectly}} = \overrightarrow{perfectly} + \overrightarrow{perfect} + \overrightarrow{ly}$), obtaining significant improvements on intrinsic evaluation tasks, word similarity and machine translation. Lample et al. (2016) concatenated character-level and word-level representations for creating word representations, and then used them as input to their models for obtaining state-of-the-art results in Named Entity Recognition on several languages.

What these works have in common is that the models they describe first learn how to represent subword information, at character (Luong and Manning, 2016), morpheme (Botha and Blunsom, 2014), or substring (Bojanowski et al., 2017) levels, and then combine these learned representations at the word level. The incorporation of information at a finer-grained hierarchy results in higher-quality modeling of rare words, morphological processes, and semantics (Avraham and Goldberg, 2017).

There is no consensus, however, on which combination method works better in which case, or how the choice of a combination method affects downstream performance, either measured intrinsically at the word level, or extrinsically at the sen-

[1] Unless using pre-trained embeddings with a notion of subword information such as `fastText` (Bojanowski et al., 2017)

tence level.

In this paper we aim to provide some intuitions about how the choice of mechanism for combining character-level with word-level representations influences the quality of the final word representations, and the subsequent effect these have in the performance of downstream tasks. Our contributions are as follows:

- We show that a feature-wise sigmoidal gating mechanism is the best at combining representations at the character and word-level hierarchies, as measured by word similarity tasks.
- We provide evidence that this mechanism learns that to properly model increasingly infrequent words, it has to increasingly rely on character-level information.
- We finally show that despite the increased expressivity of word representations it offers, it has no clear effect in sentence representations, as measured by sentence evaluation tasks.

2 Background

We are interested in studying different ways of combining word representations, obtained from different hierarchies, into a single word representation. Specifically, we want to study how combining word representations (1) taken directly from a word embedding lookup table, and (2) obtained from a function over the characters composing them, affects the quality of the final word representations.

Let $\mathcal{W}$ be a set, or vocabulary, of words with $|\mathcal{W}|$ elements, and $\mathcal{C}$ a vocabulary of characters with $|\mathcal{C}|$ elements. Further, let $\boldsymbol{x} = w_1, \ldots, w_n;\ w_i \in \mathcal{W}$ be a sequence of words, and $\boldsymbol{c}^i = c_1^i, \ldots, c_m^i;\ c_j^i \in \mathcal{C}$ be the sequence of characters composing w_i. Each token w_i can be represented as a vector $\boldsymbol{v}_i^{(w)} \in \mathbb{R}^d$ extracted directly from an embedding lookup table $\boldsymbol{E}^{(w)} \in \mathbb{R}^{|\mathcal{W}| \times d}$, pre-trained or otherwise, and as a vector $\boldsymbol{v}_i^{(c)} \in \mathbb{R}^d$ built from the characters that compose it; in other words, $\boldsymbol{v}_i^{(c)} = f(\boldsymbol{c}^i)$, where f is a function that maps a sequence of characters to a vector.

The methods for combining word and character-level representations we study, are of the form $G(\boldsymbol{v}_i^{(w)}, \boldsymbol{v}_i^{(c)}) = \boldsymbol{v}_i$ where $\boldsymbol{v}_i$ is the final word representation.

2.1 Mapping Characters to Character-level Word Representations

The function f is composed of an *embedding* layer, an optional *context* function, and an *aggregation* function.

The **embedding layer** transforms each character c_j^i into a vector $\boldsymbol{r}_j^i$ of dimension d_r, by directly taking it from a trainable embedding lookup table $\boldsymbol{E}^{(c)} \in \mathbb{R}^{|\mathcal{C}| \times d_r}$. We define the *matrix* representation of word w_i as $\boldsymbol{C}^i = [\boldsymbol{r}_1^i, \ldots, \boldsymbol{r}_m^i],\ \boldsymbol{C}^i \in \mathbb{R}^{m \times d_r}$.

The **context function** takes $\boldsymbol{C}^i$ as input and returns a context-enriched matrix representation $\boldsymbol{H}^i = [\boldsymbol{h}_1^i, \ldots, \boldsymbol{h}_m^i],\ \boldsymbol{H}^i \in \mathbb{R}^{m \times d_h}$, in which each $\boldsymbol{h}_j^i$ contains a measure of information about its context, and interactions with its neighbors. In particular, we chose to do this by feeding $\boldsymbol{C}^i$ to a Bidirectional LSTM (BiLSTM) (Graves and Schmidhuber, 2005; Graves et al., 2013)[2].

Informally, we can think of a Long Short-Term Memory Network (LSTM) (Hochreiter and Schmidhuber, 1997) as a function $\mathbb{R}^{m \times d_r} \rightarrow \mathbb{R}^{m \times d_h}$ that takes a matrix $\boldsymbol{C} = [\boldsymbol{r}_1, \ldots, \boldsymbol{r}_m]$ as input and returns a context-enriched matrix representation $\boldsymbol{H} = [\boldsymbol{h}_1, \ldots, \boldsymbol{h}_m]$, where each $\boldsymbol{h}_j$ encodes information about the previous elements $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_{j-1}$[3].

A BiLSTM is simply composed of 2 LSTMs, one that reads the input from left to right (forward), and another that does so from right to left (backward). The output of the forward and backward LSTMs are $\overrightarrow{\boldsymbol{H}} = [\overrightarrow{\boldsymbol{h}}_1, \ldots, \overrightarrow{\boldsymbol{h}}_m]$ and $\overleftarrow{\boldsymbol{H}} = [\overleftarrow{\boldsymbol{h}}_1, \ldots, \overleftarrow{\boldsymbol{h}}_m]$ respectively. In the backward case the LSTM reads $\boldsymbol{r}_m$ first and $\boldsymbol{r}_1$ last, therefore $\overleftarrow{\boldsymbol{h}}_j$ will encode the context from $\overleftarrow{\boldsymbol{h}}_{j+1}, \ldots, \overleftarrow{\boldsymbol{h}}_m$.

The **aggregation function** takes the context-enriched matrix representation of word w_i for both directions, $\overrightarrow{\boldsymbol{H}^i}$ and $\overleftarrow{\boldsymbol{H}^i}$, and returns a single vector $\boldsymbol{v}_i^{(c)} \in \mathbb{R}^{d_h}$. To do so we followed Miyamoto and Cho (2016), and defined the character-level representation $\boldsymbol{v}_i^{(c)}$ of word w_i as the linear combination of the forward and backward last hidden states re-

[2] Other methods for encoding the characters' context, such as CNNs (Kim et al., 2016), could also be used.

[3] In terms of implementation, the LSTM is applied iteratively to each element of the input sequence regardless of dimension m, which means it accepts inputs of variable length, but we will use this notation for the sake of simplicity.

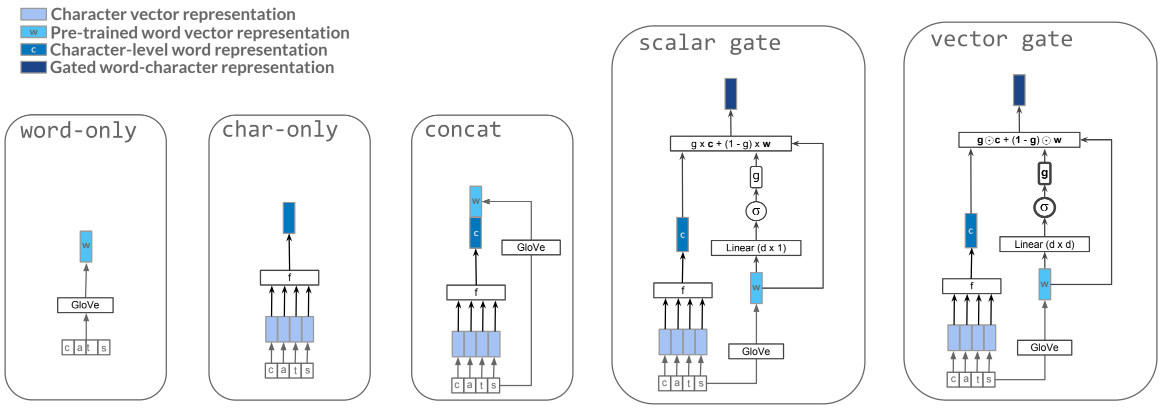

Character vector representation
w Pre-trained word vector representation
c Character-level word representation
Gated word-character representation
word-only
char-only
concat
scalar gate
vector gate
w
c
f
GloVe
c a t s
g x c + (1 - g) x w
g
σ
Linear (d x 1)
g ⊙ c + (1 - g) ⊙ w
Linear (d x d)

Figure 1: Character and Word-level combination methods.

turned by the context function:

$$\boldsymbol{v}_i^{(c)} = \boldsymbol{W}^{(c)}[\overrightarrow{\boldsymbol{h}^i}_m; \overleftarrow{\boldsymbol{h}^i}_1] + \boldsymbol{b}^{(c)} \qquad (1)$$

where $\boldsymbol{W}^{(c)} \in \mathbb{R}^{d_h \times 2d_h}$ and $\boldsymbol{b}^{(c)} \in \mathbb{R}^{d_h}$ are trainable parameters, and $[\circ; \circ]$ represents the concatenation operation between two vectors.

2.2 Combining Character and Word-level Representations

We tested three different methods for combining $\boldsymbol{v}_i^{(c)}$ with $\boldsymbol{v}_i^{(w)}$: simple concatenation, a learned scalar gate (Miyamoto and Cho, 2016), and a learned vector gate (also referred to as feature-wise sigmoidal gate). Additionally, we compared these methods to two baselines: using pre-trained word vectors only, and using character-only features for representing words. See fig. 1 for a visual description of the proposed methods.

word-only (w) considers only $\boldsymbol{v}_i^{(w)}$ and ignores $\boldsymbol{v}_i^{(c)}$:

$$\boldsymbol{v}_i = \boldsymbol{v}_i^{(w)} \qquad (2)$$

char-only (c) considers only $\boldsymbol{v}_i^{(c)}$ and ignores $\boldsymbol{v}_i^{(w)}$:

$$\boldsymbol{v}_i = \boldsymbol{v}_i^{(c)} \qquad (3)$$

concat (cat) concatenates both word and character-level representations:

$$\boldsymbol{v}_i = [\boldsymbol{v}_i^{(c)}; \boldsymbol{v}_i^{(w)}] \qquad (4)$$

scalar gate (sg) implements the scalar gating mechanism described by Miyamoto and Cho (2016):

$$g_i = \sigma(\boldsymbol{w}^\top \boldsymbol{v}_i^{(w)} + b) \qquad (5)$$

$$\boldsymbol{v}_i = g_i \boldsymbol{v}_i^{(c)} + (1 - g_i)\boldsymbol{v}_i^{(w)} \qquad (6)$$

where $\boldsymbol{w} \in \mathbb{R}^d$ and $b \in \mathbb{R}$ are trainable parameters, $g_i \in (0, 1)$, and σ is the sigmoid function.

vector gate (vg):

$$\boldsymbol{g}_i = \sigma(\boldsymbol{W}\boldsymbol{v}_i^{(w)} + \boldsymbol{b}) \qquad (7)$$

$$\boldsymbol{v}_i = \boldsymbol{g}_i \odot \boldsymbol{v}_i^{(c)} + (\boldsymbol{1} - \boldsymbol{g}_i) \odot \boldsymbol{v}_i^{(w)} \qquad (8)$$

where $\boldsymbol{W} \in \mathbb{R}^{d \times d}$ and $\boldsymbol{b} \in \mathbb{R}^d$ are trainable parameters, $\boldsymbol{g}_i \in (0, 1)^d$, σ is the element-wise sigmoid function, $\odot$ is the element-wise product for vectors, and $\boldsymbol{1} \in \mathbb{R}^d$ is a vector of ones.

The vector gate is inspired by Miyamoto and Cho (2016) and Yang et al. (2017), but is different to the former in that the gating mechanism acts upon each dimension of the word and character-level vectors, and different to the latter in that it does not rely on external sources of information for calculating the gating mechanism.

Finally, note that `word only` and `char only` are special cases of both gating mechanisms: $g_i = 0$ (scalar gate) and $\boldsymbol{g}_i = \boldsymbol{0}$ (vector gate) correspond to `word only`; $g_i = 1$ and $\boldsymbol{g}_i = \boldsymbol{1}$ correspond to `char only`.

2.3 Obtaining Sentence Representations

To enable sentence-level classification we need to obtain a sentence representation from the word vectors $\boldsymbol{v}_i$. We achieved this by using a BiLSTM with max pooling, which was shown to be a good universal sentence encoding mechanism (Conneau et al., 2017).

Let $\boldsymbol{x} = w_1, \ldots, w_n$, be an input sentence and $\boldsymbol{V} = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n]$ its matrix representation, where each $\boldsymbol{v}_i$ was obtained by one of the methods described in section 2.2. $\boldsymbol{S} = [\boldsymbol{s}_1, \ldots, \boldsymbol{s}_n]$ is the context-enriched matrix representation of $\boldsymbol{x}$ obtained by feeding $\boldsymbol{V}$ to a BiLSTM of output dimension d_s[4]. Lastly, $\boldsymbol{s} \in \mathbb{R}^{d_s}$ is the final sentence representation of $\boldsymbol{x}$ obtained by max-pooling $\boldsymbol{S}$ along the sequence dimension.

Finally, we initialized the word representations $\boldsymbol{v}_i^{(w)}$ using GloVe embeddings (Pennington et al., 2014), and fine-tuned them during training. Refer to appendix A for details on the other hyperparameters we used.

3 Experiments

3.1 Experimental Setup

We trained our models for solving the Natural Language Inference (NLI) task in two datasets, SNLI (Bowman et al., 2015) and MultiNLI (Williams et al., 2018), and validated them in each corresponding development set (including the matched and mismatched development sets of MultiNLI).

For each dataset-method combination we trained 7 models initialized with different random seeds, and saved each when it reached its best validation accuracy[5]. We then evaluated the quality of each trained model's word representations $\boldsymbol{v}_i$ in 10 word similarity tasks, using the system created by Jastrzebski et al. (2017)[6].

Finally, we fed these obtained word vectors to a BiLSTM with max-pooling and evaluated the final sentence representations in 11 downstream transfer tasks (Conneau et al., 2017; Subramanian et al., 2018).

3.2 Datasets

Word-level Semantic Similarity A desirable property of vector representations of words is that semantically similar words should have similar vector representations. Assessing whether a set of word representations possesses this quality is referred to as the semantic similarity task. This is the most widely-used evaluation method for evaluating word representations, despite its shortcomings (Faruqui et al., 2016).

This task consists of comparing the similarity between word vectors measured by a distance metric (usually cosine distance), with a similarity score obtained from human judgements. High correlation between these similarities is an indicator of good performance.

A problem with this formulation though, is that the definition of "similarity" often confounds the meaning of both *similarity* and *relatedness*. For example, *cup* and *tea* are related but dissimilar words, and this type of distinction is not always clear (Agirre et al., 2009; Hill et al., 2015).

To face the previous problem, we tested our methods in a wide variety of datasets, including some that explicitly model relatedness (WS353R), some that explicitly consider similarity (WS353S, SimLex999, SimVerb3500), and some where the distinction is not clear (MEN, MTurk287, MTurk771, RG, WS353). We also included the RareWords (RW) dataset for evaluating the quality of rare word representations. See appendix B for a more complete description of the datasets we used.

Sentence-level Evaluation Tasks Unlike word-level representations, there is no consensus on the desirable properties sentence representations should have. In response to this, Conneau et al. (2017) created SentEval[7], a sentence representation evaluation benchmark designed for assessing how well sentence representations perform in various downstream tasks (Conneau and Kiela, 2018).

Some of the datasets included in SentEval correspond to sentiment classification (CR, MPQA, MR, SST2, and SST5), subjectivity classification (SUBJ), question-type classification (TREC), recognizing textual entailment (SICK E), estimating semantic relatedness (SICK R), and measuring textual semantic similarity (STS16, STSB). The datasets are described by Conneau et al. (2017), and we provide pointers to their original sources in the appendix table B.2.

To evaluate these sentence representations SentEval trained a linear model on top of them, and evaluated their performance in the validation sets accompanying each dataset. The only exception was the STS16 task, in which our representations were evaluated directly.

4 Word-level Evaluation

4.1 Word Similarity

Table 1 shows the quality of word representations in terms of the correlation between word similarity

[4] $\boldsymbol{s}_i = [\overrightarrow{\boldsymbol{s}_i}; \overleftarrow{\boldsymbol{s}_i}]$ for each i, and both $\overrightarrow{\boldsymbol{s}_i}$ and $\overleftarrow{\boldsymbol{s}_i} \in \mathbb{R}^{\frac{d_s}{2}}$.

[5] We found that models validated on the matched development set of MultiNLI, rather than the mismatched, yielded best results, although the differences were not statistically significant.

[6] https://github.com/kudkudak/word-embeddings-benchmarks/tree/8fd0489

[7] https://github.com/facebookresearch/SentEval/tree/906b34a

		MEN	MTurk287	MTurk771	RG65	RW	SimLex999	SimVerb3500	WS353	WS353R	WS353S
SNLI	`w`	71.78	35.40	49.05	61.80	18.43	19.17	10.32	39.27	28.01	53.42
	`c`	9.85	-5.65	0.82	-5.28	17.81	0.86	2.76	-2.20	0.20	-3.87
	`cat`	71.91	**35.52**	48.84	62.12	18.46	19.10	10.21	39.35	28.16	53.40
	`sg`	70.49	34.49	46.15	59.75	18.24	17.20	8.73	35.86	23.48	50.83
	`vg`	**80.00**	32.54	**62.09**	**68.90**	**20.76**	**37.70**	**20.45**	**54.72**	**47.24**	**65.60**
MNLI	`w`	68.76	50.15	68.81	65.83	18.43	42.21	25.18	61.10	58.21	70.17
	`c`	4.84	0.06	1.95	-0.06	12.18	3.01	1.52	-4.68	-3.63	-3.65
	`cat`	68.77	50.40	68.77	65.92	18.35	42.22	25.12	61.15	58.26	70.21
	`sg`	67.66	49.58	68.29	64.84	18.36	41.81	24.57	60.13	57.09	69.41
	`vg`	**76.69**	**56.06**	**70.13**	**69.00**	**25.35**	**48.40**	**35.12**	**68.91**	**64.70**	**77.23**

Table 1: Word-level evaluation results. Each value corresponds to average Pearson correlation of 7 identical models initialized with different random seeds. Correlations were scaled to the $[-100; 100]$ range for easier reading. **Bold** values represent the best method per training dataset, per task; **underlined** values represent the best-performing method per task, independent of training dataset. For each task and dataset, every best-performing method was significantly different to other methods ($p < 0.05$), except for `w` trained in SNLI at the MTurk287 task. Statistical significance was obtained with a two-sided Welch's t-test for two independent samples without assuming equal variance (Welch, 1947).

scores obtained by the proposed models and word similarity scores defined by humans.

First, we can see that for each task, `character only` models had significantly worse performance than every other model trained on the same dataset. The most likely explanation for this is that these models are the only ones that need to learn word representations from scratch, since they have no access to the global semantic knowledge encoded by the GloVe embeddings.

Further, **bold** results show the overall trend that `vector gates` outperformed the other methods regardless of training dataset. This implies that learning how to combine character and word-level representations at the dimension level produces word vector representations that capture a notion of word similarity and relatedness that is closer to that of humans.

Additionally, results from the MNLI row in general, and **underlined** results in particular, show that training on MultiNLI produces word representations better at capturing word similarity. This is probably due to MultiNLI data being richer than that of SNLI. Indeed, MultiNLI data was gathered from various sources (novels, reports, letters, and telephone conversations, among others), rather than the single image captions dataset from which SNLI was created.

Exceptions to the previous rule are models evaluated in MEN and RW. The former case can be explained by the MEN dataset[8] containing only words that appear as image labels in the ESP-Game[9] and MIRFLICKR-1M[10] image datasets (Bruni et al., 2014), and therefore having data that is more closely distributed to SNLI than to MultiNLI.

More notably, in the RareWords dataset (Luong et al., 2013), the `word only`, `concat`, and `scalar gate` methods performed equally, despite having been trained in different datasets ($p > 0.1$), and the `char only` method performed significantly worse when trained in MultiNLI. The `vector gate`, however, performed significantly better than its counterpart trained in SNLI. These facts provide evidence that this method is capable of capturing linguistic phenomena that the other methods are unable to model.

4.2 Word Frequencies and Gating Values

Figure 2 shows that for more common words the `vector gate` mechanism tends to favor only a few dimensions while keeping a low average gating value across dimensions. On the other hand, values are greater and more homogeneous across dimensions in rarer words. Further, fig. 3 shows this mechanism assigns, on average, a greater gating value to less frequent words, confirming the findings by Miyamoto and Cho (2016), and Yang et al. (2017).

In other words, the less frequent the word, the more this mechanism allows the character-level representation to influence the final word representation, as shown by eq. (8). A possible interpretation of this result is that exploiting charac-

[8] `https://staff.fni.uva.nl/e.bruni/MEN`

[9] `http://www.cs.cmu.edu/~biglou/resources/`

[10] `http://press.liacs.nl/mirflickr/`

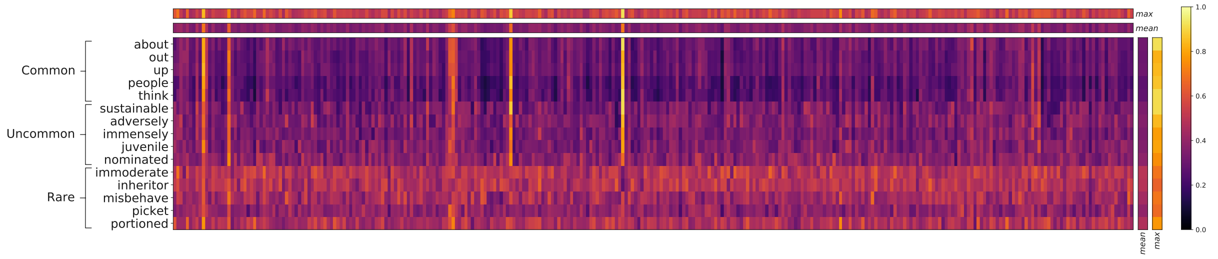


Figure 2: Visualization of gating values for 5 common words (freq. $\sim$ 20000), 5 uncommon words (freq. $\sim$ 60), and 5 rare words (freq. $\sim$ 2), appearing in both the RW and MultiNLI datasets.

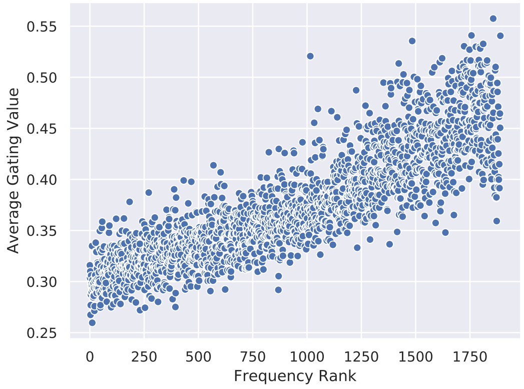


Figure 3: Average gating values for words appearing in both RW and MultiNLI. Words are sorted by decreasing frequency in MultiNLI.

ter information becomes increasingly necessary as word-level representations' quality decrease.

Another observable trend in both figures is that gating values tend to be low on average. Indeed, it is possible to see in fig. 3 that the average gating values range from 0.26 to 0.56. This result corroborates the findings by Miyamoto and Cho (2016), stating that setting $g = 0.25$ in eq. (6), was better than setting it to higher values.

In summary, the gating mechanisms learn how to compensate the lack of expressivity of underrepresented words by selectively combining their representations with those of characters.

5 Sentence-level Evaluation

Table 2 shows the impact that different methods for combining character and word-level word representations have in the quality of the sentence representations produced by our models.

We can observe the same trend mentioned in section 4.1, and highlighted by the difference between **bold** values, that models trained in MultiNLI performed better than those trained in SNLI at a statistically significant level, confirming the findings of Conneau et al. (2017). In other words, training sentence encoders on MultiNLI yields more general sentence representations than doing so on SNLI.

The two exceptions to the previous trend, SICKE and SICKR, benefited more from models trained on SNLI. We hypothesize this is again due to both SNLI and SICK (Marelli et al., 2014) having similar data distributions[11].

Additionally, there was no method that significantly outperformed the `word only` baseline in classification tasks. This means that the added expressivity offered by explicitly modeling characters, be it through concatenation or gating, was not significantly better than simply fine-tuning the pre-trained GloVe embeddings for this type of task. We hypothesize this is due to the conflation of two effects. First, the fact that morphological processes might not encode important information for solving these tasks; and second, that SNLI and MultiNLI belong to domains that are too dissimilar to the domains in which the sentence representations are being tested.

On the other hand, the `vector gate` significantly outperformed every other method in the STSB task when trained in both datasets, and in the STS16 task when trained in SNLI. This again hints at this method being capable of modeling phenomena at the word level, resulting in improved semantic representations at the sentence level.

6 Relationship Between Word- and Sentence-level Evaluation Tasks

It is clear that the better performance the `vector gate` had in word similarity tasks did not trans-

[11] SICK was created from Flickr-8k (Rashtchian et al., 2010), and SNLI from its expanded version: Flickr30k (Young et al., 2014).

		Classification							Entailment	Relatedness	Semantic Textual Similarity	
		CR	**MPQA**	**MR**	**SST2**	**SST5**	**SUBJ**	**TREC**	**SICKE**	**SICKR**†	**STS16**†	**STSB**†
SNLI	w	80.50	84.59	74.18	78.86	42.33	**90.38**	**86.83**	86.37	88.52	59.90*	71.29*
	c	74.90*	78.86*	65.93*	69.42*	35.56*	82.97*	83.31*	84.13*	83.89*	59.33*	67.20*
	cat	80.44	84.66	74.31	78.37	41.34*	90.28	85.80*	**86.40**	88.44	59.90*	71.24*
	sg	**80.59**	84.60	**74.49**	**79.04**	41.63*	90.16	86.00	86.10*	**88.57**	60.05*	71.34*
	vg	80.42	**84.66**	74.26	78.87	**42.38**	90.07	85.97	85.67	88.31*	**60.92**	**71.99**
MNLI	w	83.80	**89.13**	79.05	83.38	45.21	91.79	89.23	84.92	86.33	66.08	71.96*
	c	70.23*	72.19*	62.83*	64.55*	32.47*	79.49*	74.74*	81.53*	75.92*	51.47*	61.74*
	cat	**83.96**	89.12	**79.23**	83.70	45.08*	**91.92**	**90.03**	**85.06**	86.45	**66.17**	71.82*
	sg	83.88	89.06	79.22	83.71	45.26	91.66*	88.83*	84.96	86.40	65.49*	71.87*
	vg	83.45*	89.05	79.13	**83.87**	**45.88**	91.55*	89.49	84.82	**86.50**	65.75	**72.82**

Table 2: Experimental results. Each value shown in the table is the average result of 7 identical models initialized with different random seeds. Values represent accuracy (%) unless indicated by †, in which case they represent Pearson correlation scaled to the range $[-100, 100]$ for easier reading. **Bold** values represent the best method per training dataset, per task; **underlined** values represent the best-performing method per task, independent of training dataset. Values marked with an asterisk (*) are significantly different to the average performance of the best model trained on the same dataset ($p < 0.05$). Results for every best-performing method trained on one dataset are significantly different to the best-performing method trained on the other. Statistical significance was obtained in the same way as described in table 1.

late into overall better performance in downstream tasks. This confirms previous findings indicating that intrinsic word evaluation metrics are not good predictors of downstream performance (Tsvetkov et al., 2015; Chiu et al., 2016; Faruqui et al., 2016; Gladkova and Drozd, 2016).

Figure 4(b) shows that the word representations created by the `vector gate` trained in MultiNLI had positively-correlated results within several word-similarity tasks. This hints at the generality of the word representations created by this method when modeling similarity and relatedness.

However, the same cannot be said about sentence-level evaluation performance; there is no clear correlation between word similarity tasks and sentence-evaluation tasks. This is clearly illustrated by performance in the STSBenchmark, the only in which the `vector gate` was significantly superior, not being correlated with performance in any word-similarity dataset. This can be interpreted simply as word-level representations capturing word-similarity not being a sufficient condition for good performance in sentence-level tasks.

In general, fig. 4 shows that there are no general correlation effects spanning both training datasets and combination mechanisms. For example, fig. 4(a) shows that, for both `word-only` and `concat` models trained in SNLI, performance in word similarity tasks correlates positively with performance in most sentence evaluation tasks, however, this does not happen as clearly for the same models trained in MultiNLI (fig. 4(b)).

7 Related Work

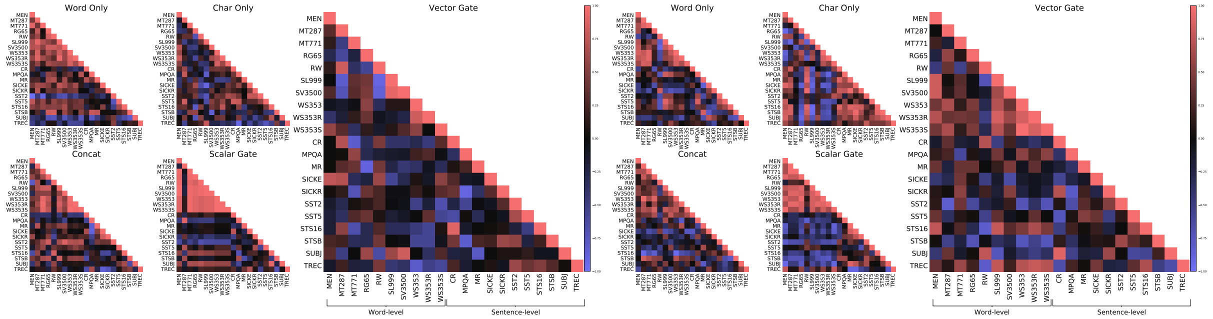
7.1 Gating Mechanisms for Combining Characters and Word Representations

To the best of our knowledge, there are only two recent works that specifically study how to combine word and subword-level vector representations.

Miyamoto and Cho (2016) propose to use a trainable scalar gating mechanism capable of learning a weighting scheme for combining character-level and word-level representations. They compared their proposed method to manually weighting both levels; using characters only; words only; or their concatenation. They found that in some datasets a specific manual weighting scheme performed better, while in others the learned scalar gate did.

Yang et al. (2017) further expand the gating concept by making the mechanism work at a finer-grained level, learning how to weight each vector's dimensions independently, conditioned on external word-level features such as part-of-speech and named-entity tags. Similarly, they compared their proposed mechanism to using words only, characters only, and a concatenation of both, with and without external features. They found that their vector gate performed better than the other methods in all the reported tasks, and beat the state of the art in two reading comprehension tasks.

Both works showed that the gating mechanisms assigned greater importance to character-level rep-



(a) Models trained in SNLI.

(b) Models trained in MultiNLI.

Figure 4: Spearman correlation between performances in word and sentence level evaluation tasks.

resentations in rare words, and to word-level representations in common ones, reaffirming the previous findings that subword structures in general, and characters in particular, are beneficial for modeling uncommon words.

7.2 Sentence Representation Learning

The problem of representing sentences as fixed-length vectors has been widely studied.

Zhao et al. (2015) suggested a self-adaptive hierarchical model that gradually composes words into intermediate phrase representations, and adaptively selects specific hierarchical levels for specific tasks. Kiros et al. (2015) proposed an encoder-decoder model trained by attempting to reconstruct the surrounding sentences of an encoded passage, in a fashion similar to Skip-gram (Mikolov et al., 2013). Hill et al. (2016) overcame the previous model's need for ordered training sentences by using autoencoders for creating the sentence representations. Jernite et al. (2017) implemented a model simpler and faster to train than the previous two, while having competitive performance. Similar to Kiros et al. (2015), Gan et al. (2017) suggested predicting future sentences with a hierarchical CNN-LSTM encoder.

Conneau et al. (2017) trained several sentence encoding architectures on a combination of the SNLI and MultiNLI datasets, and showed that a BiLSTM with max-pooling was the best at producing highly transferable sentence representations. More recently, Subramanian et al. (2018) empirically showed that sentence representations created in a multi-task setting (Collobert and Weston, 2008), performed increasingly better the more tasks they were trained in. Zhang et al. (2018) proposed using an autoencoder that relies on multi-head self-attention over the concatenation of the max and mean pooled encoder outputs for producing sentence representations. Finally, Wieting and Kiela (2019) show that modern sentence embedding methods are not vastly superior to random methods.

The works mentioned so far usually evaluate the quality of the produced sentence representations in sentence-level downstream tasks. Common benchmarks grouping these kind of tasks include SentEval (Conneau and Kiela, 2018), and GLUE (Wang et al., 2019). Another trend, however, is to *probe* sentence representations to understand what linguistic phenomena they encode (Linzen et al., 2016; Adi et al., 2017; Conneau et al., 2018; Perone et al., 2018; Zhu et al., 2018).

7.3 General Feature-wise Transformations

Dumoulin et al. (2018) provide a review on feature-wise transformation methods, of which the mechanisms presented in this paper form a part of. In a few words, the g parameter, in both `scalar gate` and `vector gate` mechanisms, can be understood as a *scaling parameter* limited to the $(0, 1)$ range and conditioned on word representations, whereas adding the scaled $\boldsymbol{v}_i^{(c)}$ and $\boldsymbol{v}_i^{(w)}$ representations can be seen as *biasing* word representations conditioned on character representations.

The previous review extends the work by Perez et al. (2018), which describes the Feature-wise Linear Modulation (FiLM) framework as a generalization of Conditional Normalization methods, and apply it in visual reasoning tasks. Some of the reported findings are that, in general, scaling has greater impact than biasing, and that in a setting similar to the `scalar gate`, limiting the scaling parameter to $(0, 1)$ hurt performance. Future decisions involving the design of mechanisms for combining character and word-level representations should be informed by these insights.

8 Conclusions

We presented an empirical study showing the effect that different ways of combining character and word representations has in word-level and sentence-level evaluation tasks.

We showed that a vector gate performed consistently better across a variety of word similarity and relatedness tasks. Additionally, despite showing inconsistent results in sentence evaluation tasks, it performed significantly better than the other methods in semantic similarity tasks.

We further showed through this mechanism, that learning character-level representations is always beneficial, and becomes increasingly so with less common words.

In the future it would be interesting to study how the choice of mechanism for combining subword and word representations affects the more recent language-model-based pretraining methods such as ELMo (Peters et al., 2018), GPT (Radford et al., 2018, 2019) and BERT (Devlin et al., 2018).

Acknowledgements

Thanks to Edison Marrese-Taylor and Pablo Loyola for their feedback on early versions of this manuscript. We also gratefully acknowledge the support of the NVIDIA Corporation with the donation of one of the GPUs used for this research. Jorge A. Balazs is partially supported by the Japanese Government MEXT Scholarship.

References

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2017. Fine-grained Analysis of Sentence Embeddings Using Auxiliary Prediction Tasks. In *Proceedings of the 5th International Conference on Learning Representations (ICLR)*, Toulon, France.

Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Pasca, 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, 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.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A Simple but Tough-to-Beat Baseline for Sentence Embeddings. In *International Conference on Learning Representations*.

Oded Avraham and Yoav Goldberg. 2017. The Interplay of Semantics and Morphology in Word Embeddings. *arXiv preprint arXiv:1704.01938*.

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.

Jan Botha and Phil Blunsom. 2014. Compositional Morphology for Word Representations and Language Modelling. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1899–1907, Bejing, China. PMLR.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A Large Annotated Corpus for Learning Natural Language Inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Elia Bruni, Nam-Khanh Tran, and Marco Baroni. 2014. Multimodal Distributional Semantics. *Journal of Artificial Intelligence Research*, 49:1–47.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo 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.

Billy Chiu, Anna Korhonen, and Sampo Pyysalo. 2016. Intrinsic Evaluation of Word Vectors Fails to Predict Extrinsic Performance. In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, pages 1–6, Berlin, Germany. Association for Computational Linguistics.

Ronan Collobert and Jason Weston. 2008. A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. In *Proceedings of the 25th Annual International Conference on Machine Learning (ICML 2008)*, pages 160–167, Helsinki, Finland.

Alexis Conneau and Douwe Kiela. 2018. SentEval: An Evaluation Toolkit for Universal Sentence Representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*, Miyazaki, Japan. European Language Resource Association.

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, Germán 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.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR*, abs/1810.04805.

Vincent Dumoulin, Ethan Perez, Nathan Schucher, Florian Strub, Harm de Vries, Aaron Courville, and Yoshua Bengio. 2018. Feature-wise transformations. *Distill*.

Manaal Faruqui, Yulia Tsvetkov, Pushpendre Rastogi, and Chris Dyer. 2016. Problems With Evaluation of Word Embeddings Using Word Similarity Tasks. In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, pages 30–35, Berlin, Germany. Association for Computational Linguistics.

Christiane Fellbaum, editor. 1998. *WordNet: an Electronic Lexical Database*. MIT Press.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2002. Placing Search in Context: The Concept Revisited. *ACM Transactions on Information Systems*, 20(1):116–131.

Zhe Gan, Yunchen Pu, Ricardo Henao, Chunyuan Li, Xiaodong He, and Lawrence Carin. 2017. Learning Generic Sentence Representations Using Convolutional Neural Networks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2390–2400, Copenhagen, Denmark. Association for Computational Linguistics.

Daniela Gerz, Ivan Vulić, Felix Hill, Roi Reichart, and Anna Korhonen. 2016. SimVerb-3500: A Large-Scale Evaluation Set of Verb Similarity. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2173–2182, Austin, Texas. Association for Computational Linguistics.

Anna Gladkova and Aleksandr Drozd. 2016. Intrinsic Evaluations of Word Embeddings: What Can We Do Better? In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, pages 36–42, Berlin, Germany. Association for Computational Linguistics.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. 2013. Speech Recognition with Deep Recurrent Neural Networks. In *Proceedings of the 2013 International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6645–6649, Vancouver, Canada. IEEE.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise Phoneme Classification with Bidirectional LSTM and Other Neural Network Architectures. *Neural Networks*, 18(5-6):602–610.

Guy Halawi, Gideon Dror, Evgeniy Gabrilovich, and Yehuda Koren. 2012. Large-scale Learning of Word Relatedness with Constraints. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pages 1406–1414, Beijing, China. ACM.

Felix Hill, Kyunghyun Cho, and Anna Korhonen. 2016. Learning Distributed Representations of Sentences from Unlabelled Data. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1367–1377, San Diego, California. 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.

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, pages 168–177, Seattle, Washington. ACM.

Stanisław Jastrzebski, Damian Leśniak, and Wojciech Marian Czarnecki. 2017. How to evaluate word embeddings? on importance of data efficiency and simple supervised tasks. *arXiv preprint arXiv:1702.02170*.

Yacine Jernite, Samuel R. Bowman, and David Sontag. 2017. Discourse-Based Objectives for Fast Unsupervised Sentence Representation Learning. *CoRR*, abs/1705.00557.

John D. Hunter. 2007. Matplotlib: A 2D Graphics Environment. *Computing in Science & Engineering*, 9(3):90–95.

Eric Jones, Travis Oliphant, Pearu Peterson, et al. 2001–. SciPy: Open source scientific tools for Python.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander Rush. 2016. Character-Aware Neural Language Models. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, pages 2741–2749, Phoenix, Arizona.

Karin Kipper, Anna Korhonen, Neville Ryant, and Martha Palmer. 2008. A large-scale classification of English verbs. *Language Resources and Evaluation*, 42(1):21–40.

Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-Thought Vectors. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 3294–3302. Curran Associates, Inc.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural Architectures for Named Entity Recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Xin Li and Dan Roth. 2002. Learning Question Classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Wang Ling, Chris Dyer, Alan W Black, Isabel Trancoso, Ramon Fermandez, Silvio Amir, Luis Marujo, and Tiago Luis. 2015. Finding Function in Form: Compositional Character Models for Open Vocabulary Word Representation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1520–1530, Lisbon, Portugal. Association for Computational Linguistics.

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(1):521–535.

Minh-Thang Luong and Christopher D. Manning. 2016. Achieving Open Vocabulary Neural Machine Translation with Hybrid Word-Character Models. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1054–1063, Berlin, Germany. Association for Computational Linguistics.

Thang Luong, Richard Socher, and Christopher D. 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.

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-2014)*, Reykjavik, Iceland. European Language Resources Association (ELRA).

Wes McKinney. 2010. Data Structures for Statistical Computing in Python. In *Proceedings of the 9th Python in Science Conference*, pages 51 – 56.

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.

Yasumasa Miyamoto and Kyunghyun Cho. 2016. Gated Word-Character Recurrent Language Model. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1992–1997, Austin, Texas. Association for Computational Linguistics.

Douglas L. Nelson, Cathy L. McEvoy, and Thomas A. Schreiber. 2004. The University of South Florida free association, rhyme, and word fragment norms. *Behavior Research Methods, Instruments, & Computers*, 36(3):402–407.

Travis E. Oliphant. 2015. *Guide to NumPy*, 2nd edition. CreateSpace Independent Publishing Platform, USA.

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)*.

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.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. In *NeurIPS Autodiff Workshop*, Long Beach, California.

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.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. 2018. FiLM: Visual Reasoning with a General Conditioning Layer. In *AAAI Conference on Artificial Intelligence*, New Orleans, Louisiana.

Christian S. Perone, Roberto Silveira, and Thomas S. Paula. 2018. Evaluation of sentence embeddings in downstream and linguistic probing tasks. *CoRR*, abs/1806.06259.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. 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.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving Language Understanding by Generative Pre-Training. Technical report, OpenAI.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners. Technical report, OpenAI.

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, pages 337–346, Hyderabad, India.

Cyrus Rashtchian, Peter Young, Micah Hodosh, and Julia Hockenmaier. 2010. Collecting Image Annotations Using Amazon's Mechanical Turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 139–147, Los Angeles, California. Association for Computational Linguistics.

Guido van Rossum. 1995. Python Tutorial. Technical Report CS-R9526, Department of Computer Science, CWI, Amsterdam, The Netherlands.

Herbert Rubenstein and John B. Goodenough. 1965. Contextual Correlates of Synonymy. *Communications of the ACM*, 8(10):627–633.

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. Association for Computational Linguistics.

Sandeep Subramanian, Adam Trischler, Yoshua Bengio, and Christopher J Pal. 2018. Learning General Purpose Distributed Sentence Representations via Large Scale Multi-task Learning. In *International Conference on Learning Representations*.

Yulia Tsvetkov, Manaal Faruqui, Wang Ling, Guillaume Lample, and Chris Dyer. 2015. Evaluation of Word Vector Representations by Subspace Alignment. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2049–2054, Lisbon, Portugal. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *Proceedings of the 7th International Conference on Learning Representations (ICLR)*, New Orleans, Louisiana.

Sida Wang and Christopher Manning. 2012. Baselines and Bigrams: Simple, Good Sentiment and Topic Classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 90–94, Jeju Island, Korea. Association for Computational Linguistics.

Michael Waskom, Olga Botvinnik, Drew O'Kane, Paul Hobson, Joel Ostblom, Saulius Lukauskas, David C Gemperline, Tom Augspurger, Yaroslav Halchenko, John B. Cole, Jordi Warmenhoven, Julian de Ruiter, Cameron Pye, Stephan Hoyer, Jake Vanderplas, Santi Villalba, Gero Kunter, Eric Quintero, Pete Bachant, Marcel Martin, Kyle Meyer, Alistair Miles, Yoav Ram, Thomas Brunner, Tal Yarkoni, Mike Lee Williams, Constantine Evans, Clark Fitzgerald, Brian, and Adel Qalieh. 2018. mwaskom/seaborn: v0.9.0 (july 2018).

Bernard Lewis Welch. 1947. The Generalization of "Student's" Problem When Several Different Population Variances are Involved. *Biometrika*, 34(1-2):28–35.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating Expressions of Opinions and Emotions in Language. *Language Resources and Evaluation*, 39(2):165–210.

John Wieting and Douwe Kiela. 2019. No Training Required: Exploring Random Encoders for Sentence Classification. In *Proceedings of the 7th International Conference on Learning Representations (ICLR)*, New Orleans, Louisiana.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. 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 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *arXiv preprint arXiv:1609.08144*.

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 *Proceedings of the 5th International Conference on Learning Representations (ICLR)*, Toulon, France.

Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. 2014. From image descriptions to visual denotations. *Transactions of the Association for Computational Linguistics*, 2:67–78.

Minghua Zhang, Yunfang Wu, Weikang Li, and Wei Li. 2018. Learning Universal Sentence Representations with Mean-Max Attention Autoencoder. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4514–4523, Brussels, Belgium. Association for Computational Linguistics.

Han Zhao, Zhengdong Lu, and Pascal Poupart. 2015. Self-Adaptive Hierarchical Sentence Model. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, pages 4069–4076, Buenos Aires, Argentina. AAAI Press.

Xunjie Zhu, Tingfeng Li, and Gerard de Melo. 2018. Exploring Semantic Properties of Sentence Embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 632–637, Melbourne, Australia. Association for Computational Linguistics.

A Hyperparameters

We only considered words that appear at least twice, for each dataset. Those that appeared only once were considered UNK. We used the Treebank Word Tokenizer as implemented in NLTK[12] for tokenizing the training and development datasets.

In the same fashion as Conneau et al. (2017), we used a batch size of 64, an SGD optmizer with an initial learning rate of 0.1, and at each epoch divided the learning rate by 5 if the validation accuracy decreased. We also used gradient clipping when gradients where > 5.

We defined character vector representations as 50-dimensional vectors randomly initialized by sampling from the uniform distribution in the $(-0.05; 0.05)$ range.

The output dimension of the character-level BiLSTM was 300 per direction, and remained of such size after combining forward and backward representations as depicted in eq. 1.

Word vector representations where initialized from the 300-dimensional GloVe vectors (Pennington et al., 2014), trained in 840B tokens from the Common Crawl[13], and finetuned during training. Words not present in the GloVe vocabulary where randomly initialized by sampling from the uniform distribution in the $(-0.05; 0.05)$ range.

The input size of the word-level LSTM was 300 for every method except `concat` in which it was 600, and its output was always 2048 per direction, resulting in a 4096-dimensional sentence representation.

B Datasets

B.1 Word Similarity

Table B.1 lists the word-similarity datasets and their corresponding reference. As mentioned in section 3.2, all the word-similarity datasets contain pairs of words annotated with similarity or relatedness scores, although this difference is not always explicit. Below we provide some details for each.

MEN contains 3000 annotated word pairs with integer scores ranging from 0 to 50. Words correspond to image labels appearing in the ESP-Game[14] and MIRFLICKR-1M[15] image datasets.

MTurk287 contains 287 annotated pairs with scores ranging from 1.0 to 5.0. It was created from words appearing in both DBpedia and in news articles from The New York Times.

[12] https://www.nltk.org/

[13] https://nlp.stanford.edu/projects/glove/

[14] http://www.cs.cmu.edu/~biglou/resources/

[15] http://press.liacs.nl/mirflickr/

Dataset	Reference	URL
MEN	Bruni et al. (2014)	https://staff.fnwi.uva.nl/e.bruni/MEN
MTurk287	Radinsky et al. (2011)	https://git.io/fhQA8 (Unofficial)
MTurk771	Halawi et al. (2012)	http://www2.mta.ac.il/~gideon/mturk771.html
RG	Rubenstein and Goodenough (1965)	https://git.io/fhQAB (Unofficial)
RareWords (RW)	Luong et al. (2013)	https://nlp.stanford.edu/~lmthang/morphoNLM/
SimLex999	Hill et al. (2015)	https://fh295.github.io/simlex.html
SimVerb3500	Gerz et al. (2016)	http://people.ds.cam.ac.uk/dsg40/simverb.html
WS353	Finkelstein et al. (2002)	http://www.cs.technion.ac.il/~gabr/resources/data/wordsim353/
WS353R	Agirre et al. (2009)	http://alfonseca.org/eng/research/wordsim353.html
WS353S	Agirre et al. (2009)	http://alfonseca.org/eng/research/wordsim353.html

Table B.1: Word similarity and relatedness datasets.

MTurk771 contains 771 annotated pairs with scores ranging from 1.0 to 5.0, with words having synonymy, holonymy or meronymy relationships sampled from WordNet (Fellbaum, 1998).

RG contains 65 annotated pairs with scores ranging from 0.0 to 4.0 representing "similarity of meaning".

RW contains 2034 pairs of words annotated with similarity scores in a scale from 0 to 10. The words included in this dataset were obtained from Wikipedia based on their frequency, and later filtered depending on their WordNet synsets, including synonymy, hyperonymy, hyponymy, holonymy and meronymy. This dataset was created with the purpose of testing how well models can represent rare and complex words.

SimLex999 contains 999 word pairs annotated with similarity scores ranging from 0 to 10. In this case the authors explicitly considered similarity and not relatedness, addressing the shortcomings of datasets that do not, such as MEN and WS353. Words include nouns, adjectives and verbs.

SimVerb3500 contains 3500 verb pairs annotated with similarity scores ranging from 0 to 10. Verbs were obtained from the USF free association database (Nelson et al., 2004), and VerbNet (Kipper et al., 2008). This dataset was created to address the lack of representativity of verbs in SimLex999, and the fact that, at the time of creation, the best performing models had already surpassed inter-annotator agreement in verb similarity evaluation resources. Like SimLex999, this dataset also explicitly considers similarity as opposed to relatedness.

WS353 contains 353 word pairs annotated with similarity scores from 0 to 10.

WS353R is a subset of WS353 containing 252 word pairs annotated with relatedness scores. This dataset was created by asking humans to classify each WS353 word pair into one of the following classes: synonyms, antonyms, identical, hyperonym-hyponym, hyponym-hyperonym, holonym-meronym, meronym-holonym, and none-of-the-above. These annotations were later used to group the pairs into: *similar* pairs (synonyms, antonyms, identical, hyperonym-hyponym, and hyponym-hyperonym), *related* pairs (holonym-meronym, meronym-holonym, and none-of-the-above with a human similarity score greater than 5), and *unrelated* pairs (classified as none-of-the-above with a similarity score less than or equal to 5). This dataset is composed by the union of related and unrelated pairs.

WS353S is another subset of WS353 containing 203 word pairs annotated with similarity scores. This dataset is composed by the union of similar and unrelated pairs, as described previously.

B.2 Sentence Evaluation Datasets

Table B.2 lists the sentence-level evaluation datasets used in this paper. The provided URLs correspond to the original sources, and not necessarily to the URLs where SentEval[16] got the data from[17].

The version of the CR, MPQA, MR, and SUBJ datasets used in this paper were the ones preprocessed by Wang and Manning (2012)[18]. Both SST2 and SST5 correspond to preprocessed versions of the Stanford Sentiment Treebank (SST) dataset by Socher et al. (2013)[19]. SST2 corresponds to a subset of SST used by Arora et al. (2017) containing flat representations of sentences annotated with binary sentiment labels, and SST5 to another subset annotated with more fine-grained sentiment labels (very negative, negative, neutral, positive, very positive).

[16] https://github.com/facebookresearch/SentEval/tree/906b34a

[17] A list of the data used by SentEval can be found in its data setup script: https://git.io/fhQpq

[18] https://nlp.stanford.edu/~sidaw/home/projects:nbsvm

[19] https://nlp.stanford.edu/sentiment/

Dataset	Reference	URL
CR	Hu and Liu (2004)	`https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html#datasets`
MPQA	Wiebe et al. (2005)	`https://mpqa.cs.pitt.edu/corpora/mpqa_corpus/`
MR	Pang and Lee (2005)	`http://www.cs.cornell.edu/people/pabo/movie-review-data/`
SST2	Arora et al. (2017)	`https://github.com/PrincetonML/SIF/tree/master/data`
SST5	See caption.	`https://git.io/fhQAV`
SUBJ	Pang and Lee (2004)	`http://www.cs.cornell.edu/people/pabo/movie-review-data/`
TREC	Li and Roth (2002)	`http://cogcomp.org/Data/QA/QC/`
SICKE	Marelli et al. (2014)	`http://clic.cimec.unitn.it/composes/sick.html`
SICKR	Marelli et al. (2014)	`http://clic.cimec.unitn.it/composes/sick.html`
STS16	Agirre et al. (2016)	`http://ixa2.si.ehu.es/stswiki/index.php/Main_Page`
STSB	Cer et al. (2017)	`http://ixa2.si.ehu.es/stswiki/index.php/STSbenchmark`

Table B.2: Sentence representation evaluation datasets. SST5 was obtained from a GitHub repository with no associated peer-reviewed work.