Neural Paraphrase Identification of Questions with Noisy Pretraining

Gaurav Singh Tomar Thyago Duque Oscar Täckström
Jakob Uszkoreit Dipanjan Das
Google Inc.
{gtomar, duque, oscart, uszkoreit, dipanjand}@google.com

Abstract

We present a solution to the problem of paraphrase identification of questions. We focus on a recent dataset of question pairs annotated with binary paraphrase labels and show that a variant of the decomposable attention model (Parikh et al., 2016) results in accurate performance on this task, while being far simpler than many competing neural architectures. Furthermore, when the model is pretrained on a noisy dataset of automatically collected question paraphrases, it obtains the best reported performance on the dataset.

1 Introduction

Question paraphrase identification is a widely useful NLP application. For example, in question-and-answer (QA) forums ubiquitous on the Web, there are vast numbers of duplicate questions. Identifying these duplicates and consolidating their answers increases the efficiency of such QA forums. Moreover, identifying questions with the same semantic content could help Web-scale question answering systems that are increasingly concentrating on retrieving focused answers to users' queries.

Here, we focus on a recent dataset published by the QA website Quora.com containing over 400K annotated question pairs containing binary paraphrase labels.[1] We believe that this dataset presents a great opportunity to the NLP research community and practitioners due to its scale and quality; it can result in systems that accurately identify duplicate questions, thus increasing the quality of many QA forums. We examine a simple model family, the *decomposable attention model* of Parikh et al. (2016), that has shown promise in modeling natural language inference and has inspired recent work on similar tasks (Chen et al., 2016; Kim et al., 2017).

We present two contributions. First, to mitigate data sparsity, we modify the input representation of the decomposable attention model to use sums of character n-gram embeddings instead of word embeddings. We show that this model trained on the Quora dataset produces comparable or better results with respect to several complex neural architectures, all using pretrained word embeddings. Second, to significantly improve our model performance, we pretrain *all* our model parameters on the noisy, automatically collected question-paraphrase corpus Paralex (Fader et al., 2013), followed by fine-tuning the parameters on the Quora dataset. This two-stage training procedure achieves the best result on the Quora dataset to date, and is also significantly better than learning *only* the character n-gram embeddings during the pretraining stage.

2 Related Work

Paraphrase identification is a well-studied task in NLP (Das and Smith, 2009; Chang et al., 2010; He et al., 2015; Wang et al., 2016, *inter alia*). Here, we focus on an instance, that of finding questions with identical meaning. Lei et al. (2016) consider a related task leveraging the AskUbuntu corpus (dos Santos et al., 2015), but it contains two orders of magnitude less annotations, thus limiting the quality of any model. Most relevant to this work is that of Wang et al. (2017), who present the best results on the Quora dataset prior to this work. The *bilateral multi-perspective matching* model (BIMPM) of Wang et al. uses a character-based LSTM (Hochreiter and Schmidhuber, 1997) at its input representation layer, a layer of bi-LSTMs for computing context information, four different types of multi-perspective matching layers, an additional bi-LSTM aggregation layer, followed by a

[1] See https://data.quora.com/First-Quora-Dataset-Release-Question-Pairs.

Proceedings of the First Workshop on Subword and Character Level Models in NLP, pages 142–147,
Copenhagen, Denmark, September 7, 2017.

two-layer feedforward network for prediction. In contrast, the decomposable attention model uses four simple feedforward networks to (self-)attend, compare and predict, leading to a more efficient architecture. BIMPM falls short of our best performing model pretrained on noisy paraphrase data and uses more parameters than our best model.

Character-level modeling of text is a popular approach. While conceptually simple, character n-gram embeddings are a highly competitive representation (Huang et al., 2013; Wieting et al., 2016; Bojanowski et al., 2016). More complex representations built directly from individual characters have also been proposed (Sennrich et al., 2016; Luong and Manning, 2016; Kim et al., 2016; Chung et al., 2016; Ling et al., 2015). These representations are robust to out-of-vocabulary items, often producing improved results. Our pretraining procedure is reminiscent of several recent papers (Wieting et al., 2016, *inter alia*) who aim for general purpose character n-gram embeddings. In contrast, we pretrain all model parameters on automatic but in-domain paraphrase data. We employ the same neural architecture as our end task, similar to prior work on multi-task learning (Søgaard and Goldberg, 2016, *inter alia*), but use a simpler learning setup.

3 Approach

Our starting point is the decomposable attention model (Parikh et al., 2016, DECATT henceforth), which despite its simplicity and efficiency has been shown to work remarkably well for the related task of natural language inference (Bowman et al., 2015). We extend this model with character n-gram embeddings and noisy pretraining for the task of question paraphrase identification.

3.1 Problem Formulation

Let $\mathbf{a} = (a_1, \ldots, a_{\ell_a})$ and $\mathbf{b} = (b_1, \ldots, b_{\ell_b})$ be two input texts consisting of ℓ_a and ℓ_b tokens, respectively. We assume that each a_i, $b_j \in \mathbb{R}^d$ is encoded in a vector of dimension d. A context window of size c is subsequently applied, such that the input to the model $(\bar{\mathbf{a}}, \bar{\mathbf{b}})$ consists of partly overlapping phrases $\bar{a}_i = [a_{i-c}, \ldots, a_i, \ldots, a_{i+c}], \bar{b}_j = [b_{j-c}, \ldots, b_j, \ldots, b_{j+c}] \in \mathbb{R}^{2c+1 \times d}$. The model is estimated using training data in the form of labeled pairs $\{\mathbf{a}^{(n)}, \mathbf{b}^{(n)}, \mathbf{y}^{(n)}\}_{n=1}^{N}$, where $\mathbf{y}^{(n)} \in \{0, 1\}$ is a binary label indicating whether $\mathbf{a}$ is a paraphrase of $\mathbf{b}$ or not. Our goal is to predict the correct label $\mathbf{y}$ given a pair of previously unseen texts $(\mathbf{a}, \mathbf{b})$.

3.2 The Decomposable Attention Model

The DECATT model divides the prediction into three steps: *Attend*, *Compare* and *Aggregate*. Due to lack of space, we only provide a brief outline below and refer to Parikh et al. (2016) for further details on each of these steps.

Attend. First, the elements of $\bar{\mathbf{a}}$ and $\bar{\mathbf{b}}$ are aligned using a variant of neural attention (Bahdanau et al., 2015) to decompose the problem into the comparison of aligned phrases.

$$e_{ij} := F(\bar{a}_i)^\top F(\bar{b}_j)\,. \tag{1}$$

The function F is a feedforward network. The aligned phrases are computed as follows:

$$\beta_i := \sum_{j=1}^{\ell_b} \frac{\exp(e_{ij})}{\sum_{k=1}^{\ell_b} \exp(e_{ik})} \bar{b}_j\,,$$
$$\alpha_j := \sum_{i=1}^{\ell_a} \frac{\exp(e_{ij})}{\sum_{k=1}^{\ell_a} \exp(e_{kj})} \bar{a}_i\,. \tag{2}$$

Here β_i is the subphrase in $\bar{\mathbf{b}}$ that is (softly) aligned to $\bar{a}_i$ and vice versa for α_j. Optionally, the inputs $\bar{\mathbf{a}}$ and $\bar{\mathbf{b}}$ to (1) can be replaced by input representations passed through a "self-attention" step to capture longer context. In this optional step, we modify the input representations using "self-attention" to encode compositional relationships between words within each sentence, as proposed by (Cheng et al., 2016). Similar to (1), we define

$$f_{ij} := F_{self}(\bar{a}_i)^\top F'_{self}(\bar{a}_j)\,. \tag{3}$$

The function F_{self} and F'_{self} are feedforward networks. The self-aligned phrases are then computed as follows:

$$a'_i := \sum_{j=1}^{\ell_a} \frac{\exp(f_{ij} + d_{i-j})}{\sum_{k=1}^{\ell_a} \exp(f_{ik} + d_{i-k})} a_j\,. \tag{4}$$

where d_{i-j} is a learned distance-sensitive bias term. Subsequent steps then use modified input representations defined as $\bar{a}_i := [a_i, a'_i]$ and $\bar{b}_i := [b_i, b'_i]$.

Compare. Second, we separately compare the aligned phrases $\{(\bar{a}_i, \beta_i)\}_{i=1}^{\ell_a}$ and $\{(\bar{b}_j, \alpha_j)\}_{j=1}^{\ell_b}$ using a feedforward network G:

$$\mathbf{v}_{1,i} := G([\bar{a}_i, \beta_i]) \quad \forall i \in \langle 1, \ldots, \ell_a \rangle\,,$$
$$\mathbf{v}_{2,j} := G([\bar{b}_j, \alpha_j]) \quad \forall j \in \langle 1, \ldots, \ell_b \rangle\,. \tag{5}$$

where the brackets $[\cdot, \cdot]$ denote concatenation.

Aggregate. Finally, the sets $\{\mathbf{v}_{1,i}\}_{i=1}^{\ell_a}$ and $\{\mathbf{v}_{2,j}\}_{j=1}^{\ell_b}$ are aggregated by summation. The sum of two sets is concatenated and passed through another feedforward network followed by a linear layer, to predict the label $\hat{y}$.

3.3 Character n-Gram Word Encodings

Parikh et al. assume that each token a_i, $b_j \in \mathbb{R}^d$ is directly embedded in a vector of dimension d; in practice, they used pretrained word embeddings. Inspired by prior work mentioned in Section 2, we use an alternative approach and instead represent each token as a sum of its embedded character n-grams. This allows for more effective parameter sharing at a small additional computational cost. As observed in Section 4, this leads to better results compared to word embeddings.

3.4 Noisy Pretraining

While character n-gram encodings help in effective parameter sharing, data sparsity remains an issue. Pretraining embeddings with a task-agnostic objective on large-scale corpora (Pennington et al., 2014) is a common remedy to this problem. However, such pretraining is limited in the following ways. First, it only applies to the input representation, leaving subsequent parts of the model to random initialization. Second, there may be a domain mismatch unless embeddings are pretrained on the same domain as the end task (e.g., questions). Finally, since the objective used for pretraining differs from that of the end task (e.g., paraphrase identification), the embeddings may be suboptimal.

As an alternative to task-agnostic pretraining of embeddings on a very large corpus, we propose to pretrain all parameters of the model on a modest-sized corpus of automatically gathered, and therefore noisy examples, drawn from a similar domain.[2] As observed in Section 4, such noisy pretraining of the full model results in more accurate performance compared to using pretrained embeddings, as well as compared to only pretraining embeddings on the noisy in-domain corpus.[3]

[2]Paralex is gathered from WikiAnswers, a QA forum.

[3]The Quora data is similar to the Paralex corpus and we exploit this by pretraining our entire model on the latter. It can be argued that not all sentence pair modeling tasks may benefit similarly from the Paralex corpus and a detailed empirical study is warranted to investigate that; in this work, we restrict our scope to only the question paraphrase identification task, a very useful NLP application by itself.

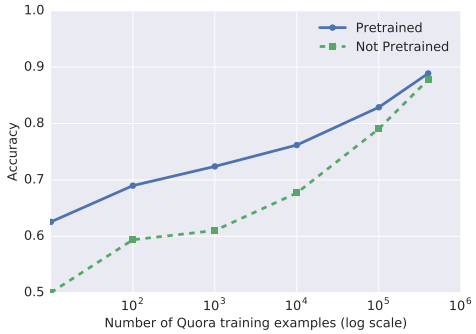

Figure 1: Learning curves for the Quora development set with and without pretraining on Paralex.

4 Experiments

4.1 Implementation Details

Datasets We evaluate our models on the Quora question paraphrase dataset which contains over 400,000 question pairs with binary labels. We use the same data and split as Wang et al. (2017), with 10,000 question pairs each for development and test, who also provide preprocessed and tokenized question pairs.[4] We duplicated the training set, which has approximately 36% positive and 64% negative pairs, by adding question pairs in reverse order (since our model is not symmetric). When pretraining the full model parameters, we use the Paralex corpus (Fader et al., 2013), which consists of 36 million noisy paraphrase pairs including duplicate reversed paraphrases. We created 64 million artificial negative paraphrase pairs (reflecting the class balance of the Quora training set) by combining the following three types of negatives in equal proportions: (1) random unrelated questions, (2) random questions that share a single word, and (3) random questions that share all but one word.[5]

Hyperparameters We tuned the following hyperparameters by grid search on the development set (settings for our best model are in parenthesis): embedding dimension (300), shape of all feedforward networks (two layers with 400 and 200 width), character n-gram sizes (5), context size (1), learning rate (0.1 for both pretraining and tuning), batch size (256 for pretraining and 64 for tuning), dropout ratio (0.1 for tuning) and prediction threshold (positive paraphrase for a score ≥ 0.3). We examined whether self-attention helps or not for all model variants, and found that it does for our best model. Note that we tried multiple orders of character n-

[4]This split is available at https://zhiguowang.github.io.

[5]More complex sampling procedures are possible, for example, by using pretrained word embeddings.

ID	Question 1	Question 2	DECATT$_{glove}$	DECATT$_{char}$	pt-DECATT$_{char}$	Gold
A	How shall I start my preparation for IIT-JEE in class 10?	Should I start preparing for the IIT JEE in class 10 only?	N	**Y**	**Y**	**Y**
B	What is fama french three factor model?	What is Fama-French three factor model?	N	**Y**	**Y**	**Y**
C	How does PayPal work in India?	Does PayPal work in India? What features of PayPal are available in India?	Y	Y	**N**	**N**
D	What are the similarities between British English and American English?	What are the similarities between American English and British English?	N	N	**Y**	**Y**
E	How is buying land on the moon a good investment? Why do people buy land on the moon?	At $20 an acre, isn't buying moon plots a solid investment?	N	N	N	**Y**
F	What can wrestlers do to prevent cauliflower ears?	Why do wrestlers have deformed ears?	N	N	N	**Y**

Table 1: Example wins and losses from the DECATT$_{glove}$, DECATT$_{char}$ and the pt-DECATT$_{char}$ models.

Method	Dev Acc	Test Acc
Siamese-CNN	-	79.60
Multi-Perspective CNN	-	81.38
Siamese-LSTM	-	82.58
Multi-Perspective-LSTM	-	83.21
L.D.C	-	85.55
BIMPM	88.69	88.17
FFNN$_{word}$	85.07	84.35
FFNN$_{char}$	86.01	85.06
DECATT$_{word}$	86.04	85.27
DECATT$_{glove}$	87.42	86.52
DECATT$_{char}$	87.78	86.84
DECATT$_{paralex-char}$	87.80	87.77
pt-DECATT$_{word}$	88.44	87.54
pt-DECATT$_{char}$	**88.89**	**88.40**

Table 2: Results on the Quora development and test sets in terms of accuracy. The first six rows are taken from (Wang et al., 2017).

grams with n $\in \{3, 4, 5\}$ both individually and separately but 5-grams alone worked better than these alternatives.

Baselines We implemented several baseline models. In our first two baselines, each question is represented by concatenating the sum of its unigram word embeddings and the sum of its trigram vectors, where each trigram vector is a concatenation of 3 adjacent word embeddings. The two question representations are then concatenated and fed to a feedforward network of shape [800, 400, 200]. We call these FFNN$_{word}$ and FFNN$_{char}$; in the latter, word embeddings are just sums of character n-gram embeddings. Second, we compare purely supervised variants of decomposable attention model, namely a word-based model without any pretrained embeddings (DECATT$_{word}$), a word-based model with GloVe (Pennington et al., 2014) embeddings (DECATT$_{glove}$), a character n-gram model (DECATT$_{char}$) without pretrained embeddings and DECATT$_{paralex-char}$ whose character n-gram embeddings are pretrained with Paralex while all other parameters are learned from scratch on Quora. Finally we present a baseline where a word-based model is pretrained completely on Paralex (pt-DECATT$_{word}$) and our best model which is a character n-gram model pretrained completely on Paralex (pt-DECATT$_{char}$). Note that in case of character n-gram based models, for tokens shorter than n characters, we backoff and emit the token itself. Also, boundary markers were added at the beginning and end of each word.

4.2 Results

Other than our baselines, we compare with Wang et al. (2017) in Table 2. We observe that the simple FFNN baselines work better than more complex Siamese and Multi-Perspective CNN or LSTM models, more so if character n-gram based embeddings are used. Our basic decomposable attention model DECATT$_{word}$ without pre-trained embeddings is better than most of the models, all of which used GloVe embeddings. An interesting observation is that DECATT$_{char}$ model without any pretrained embeddings outperforms DECATT$_{glove}$ that uses task-agnostic GloVe embeddings. Furthermore, when character n-gram embeddings are pre-trained in a task-specific manner in DECATT$_{paralex-char}$ model, we observe a significant boost in performance. [6]

The final two rows of the table show results achieved by pt-DECATT$_{word}$ and pt-DECATT$_{char}$.

[6]Note that Paralex is orders of magnitude smaller than the corpus used to pretrain GloVe.

We note that the former falls short of the DECATT$_{\text{paralex-char}}$, which shows that character n-gram representations are powerful. Finally, we note that our best performing model is pt-DECATT$_{\text{char}}$, which leverages the full power of character embeddings and pretraining the model on Paralex.

Noisy pretraining gives more significant gains in case of smaller human annotated data as can be seen in Figure 1 where non-pretrained DECATT$_{\text{char}}$ and pretrained pt-DECATT$_{\text{char}}$ are compared on a logarithmic scale of number of Quora examples. It also gives an important insight into trade off between having more but costly human annotated data versus cheap but noisy pretraining. Table 1 shows some example predictions from the DECATT$_{\text{glove}}$, DECATT$_{\text{char}}$ and the pt-DECATT$_{\text{char}}$ models. The GloVe-trained model often makes mistakes related to spelling and tokenization artifacts. We observed that hyperparameter tuning resulted in settings where non-pretrained models did not use self-attention while the pretrained character based model did, thus learning better long term context at its input layer; this is reflected in example D which shows an alternation that our best model captures. Finally, E and F show pairs that present complex paraphrases that none of our models capture.

5 Conclusion and Future Work

We presented a focused contribution on question paraphrase identification, on the recently published Quora corpus. First, we showed that replacing the word embeddings of the decomposable attention model of Parikh et al. (2016) with character n-gram embeddings results in significantly better accuracy on this task. Second, we showed that pretraining the full model on automatically labeled noisy, but task-specific data results in further improvements. Our methods perform better than several complex neural architectures and achieve state of the art. While conceptually simple, we believe that these are two important insights that may be more widely applicable within the field of natural language understanding. We leave investigation of this claim to future work that may involve evaluation on related tasks such as recognizing textual entailment.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of ICLR*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv* 1607.04606.

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

Ming-Wei Chang, Dan Goldwasser, Dan Roth, and Vivek Srikumar. 2010. Discriminative learning over constrained latent representations. In *Proceedings of HLT-NAACL*.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, and Hui Jiang. 2016. Enhancing and combining sequential and tree LSTM for natural language inference. *arXiv 1609.06038* .

Jianpeng Cheng, Li Dong, and Mirella Lapata. 2016. Long short-term memory-networks for machine reading. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Austin, Texas, pages 551–561. https://aclweb.org/anthology/D16-1053.

Junyoung Chung, Kyunghyun Cho, and Yoshua Bengio. 2016. A character-level decoder without explicit segmentation for neural machine translation. In *Proceedings of ACL*.

Dipanjan Das and Noah A. Smith. 2009. Paraphrase identification as probabilistic quasi-synchronous recognition. In *Proceedings of ACL-IJCNLP*.

Cicero dos Santos, Luciano Barbosa, Dasha Bogdanova, and Bianca Zadrozny. 2015. Learning hybrid representations to retrieve semantically equivalent questions. In *Proceedings of ACL*.

Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2013. Paraphrase-driven learning for open question answering. In *Proceedings of ACL*.

Hua He, Kevin Gimpel, and Jimmy Lin. 2015. Multi-perspective sentence similarity modeling with convolutional neural networks. In *Proceedings of EMNLP*.

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

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of CIKM*.

Yoon Kim, Carl Denton, Loung Hoang, and Alexander M. Rush. 2017. Neural machine translation by jointly learning to align and translate. In *Proceedings of ICLR*.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M. Rush. 2016. Character-aware neural language models. In *Proceedings of AAAI*.

Tao Lei, Hrishikesh Joshi, Regina Barzilay, Tommi Jaakkola, Kateryna Tymoshenko, Alessandro Moschitti, and Lluís Màrquez. 2016. Semi-supervised question retrieval with gated convolutions. In *Proceedings of NAACL*.

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 EMNLP*.

Minh-Thang Luong and Christopher D. Manning. 2016. Achieving open vocabulary neural machine translation with hybrid word-character models. In *Proceedings of ACL*.

Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. In *Proceedings of EMNLP*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of EMNLP*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of ACL*.

Anders Søgaard and Yoav Goldberg. 2016. Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of ACL*.

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017. Bilateral multi-perspective matching for natural language sentences. In *Proceedings of IJCAI*.

Zhiguo Wang, Haitao Mi, and Abraham Ittycheriah. 2016. Sentence similarity learning by lexical decomposition and composition. In *Proceedings of COLING*.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2016. Charagram: Embedding words and sentences via character n-grams. In *Proceedings of EMNLP*.