CS230

Duplicate Question Detection

Doug Chang*
Department of Computer Science
Stanford University
dougc@stanford.edu

Shiyuan Gu
Department of Computer Science
Stanford University
shgu@stanford.edu

Abstract

Duplicate question detection is one of the simplest of question answering tasks which defines a measure of similarity. The goal of this work is to establish the advantages and limitations of this similarity and to propose a framework along with additional tasks related to duplicate question detection as a preliminary analysis for defining a data pipeline for question data processing.

1 Introduction

There are many data sources for question answer type services including twitter, slack, quora, and many others. Question and answer type responses have an advantage over key word search as it is more specific and more efficient in an exchange of information. Duplicate question detection is important in establishing similarity metrics between questions and to provide consistent responses to duplicate questions in any type of question answer type service. Current measures like Jaccard Similarity and tf-idf are insufficient for questions but are more effective for bags of words approaches. Word2vec[16] is a word embedding which captures similarity between words but does not take into account dependencies outside of neighboring words.

2 Related work

Similar tasks to duplicate sentence detection include Sentiment Classification and Recognizing Text Entailment(RTE). Liu[1] shows the advantage of inner attention on RTE using RNNs/LSTMs/GRUs in a Siamese Network to model the long term dependencies beyond the neighboring word dependencies provided by wordvec models such as GloVe. Liu[1] reports 85 percent test accuracy. The SNLI dataset used by Liu[1] and others in the RTE task contains mostly single sentences. The quora dataset consists of a multimodal distribution with 300k single sentence pairs. The distribution of longer sentences is shown in table X. A question in a pair with more than 1 sentence which would make these questions documents. Document and sentence similarity are different tasks requiring different architectures for optimal performance. Deeper networks would be more useful in optimizing for multisentence documents while wider more shallow networks would have an advantage over deeper networks for single sentence similarity. Infersent[5] uses max pooling with a BiLSTM without any attention mechanisms to achieve the best performance for RTE using the SNLI dataset. One contribution from Infersent is to concatenate sentences by taking the differences and addition of the embedding vectors.

*

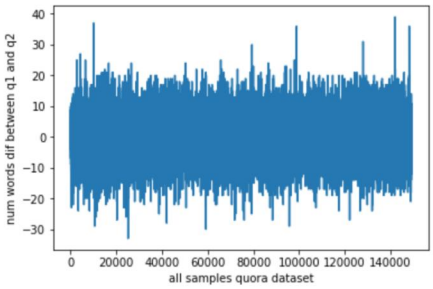
CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

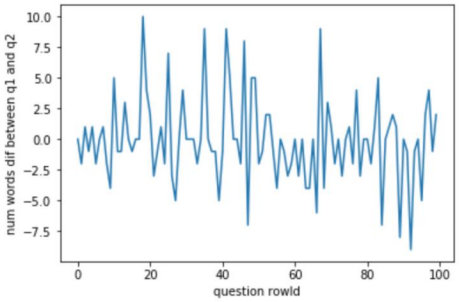
2.1 Document Similarity

Yang[3] uses datasets with longer sentences which resemble documents. The dataset consists of 8.5M documents from Yelp reviews, IMDB reviews, Yahoo Answers and Amazon Reviews. The sentence length varies from 6.4 to 14. A hierarchial network with attention mechanisms at both the word level and sentence level makes sense for learning deeper representations for documents with many sentences.

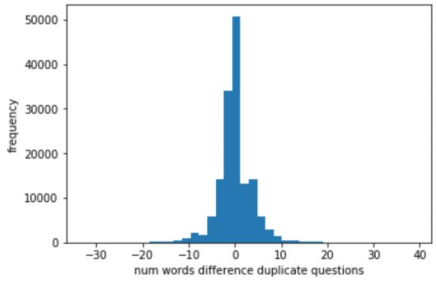
3 Dataset and Features

We used the quora dataset[15] for duplicate questions. This dataset consists of both "short" questions and "longer" questions which we define as questions having more than 1 sentence. A question with more than 1 sentence is a document. We graph the statistics of the Quora dataset to describe the dataset bias. The graph below shows the number of word difference between questions. Longer questions have a smaller chance of being a duplicate because there is a smaller probability the additional information contained in the additional words will match. The graph of the complete dataset shows the variation in number of words between question pairs. The word difference feature is an indicator of whether 2 questions are similar. Some questions with a large number of word difference include a question in a sentence paired with a single word question "What?" This obviously is not a duplicate.





The graph in Figure below is a magnified scale showing the number of word differences spike at 10 and -8. The graph shows the number of word differences between questions which are duplicates. Duplicate questions don't have a word difference of more than 10 words between the 2 questions. This can be used as a feature to improve classifier performance.



num sentences	question 1	question 2	num_sentences		
0	376717	380851	8	17	8
1	18615	16396	9	21	6
2	7040	5279	10	5	2
3	1615	1117	11	3	1
4	550	400	12	1	1
5	162	95	13	2	-
6	92	61	18	1	-
7	36	15	21	1	-

It would be easy for a classifier to look at the number of words difference as a feature. If the difference is more than 10 there is a good chance it is not a duplicate. Duplicate questions show a word length difference of ± 10 words. We eliminate questions more than a difference of +-10 and see if this set performs better on the shorter architectures like biLSTM w/max pooling and Inner Attention. We use pre-trained word vectors for our models [18]. The Quora dataset has 400k rows, we randomly split into 80% for training and 20% for testing.

1614	How often should I massage my face with Argan oil? Apparently, Argan oil is extremely beneficial for acne scars, healthy/radiant skins and etc. How often should I massage it onto my face before going to bed? Could I maybe use Argan oil one night and then the Liz Earle repair moisturiser the next to achieve a healthy, young and radiant skin? Plz help with answering :).	Which one should I use: olive oil or coconut oil for acne?	0
1171	Have you ever encountered any alien? Please be true.Do not make a story.	Have you encountered an alien? What happened?	1
979	How do I install Cydia without jailbreak on iOS 9.3.5.?	Can you download Cydia on your device without jailbreaking it?	0
	How can I increase the speed of my internet co...	How can Internet speed be increased by hacking...	0

3.1 Ambiguity

There are substantial amount of ambiguity in duplicate question pairings. At this time we do not have a way to quantify or how to automatically identify these using NLP and how to measure this effect on the final accuracy. We put a few question pairs above. The first question pair shows a big difference in word length leads to a no duplicate bias and the first passages in blue is a duplicate but the second one is not. It can be argued this is not correct.

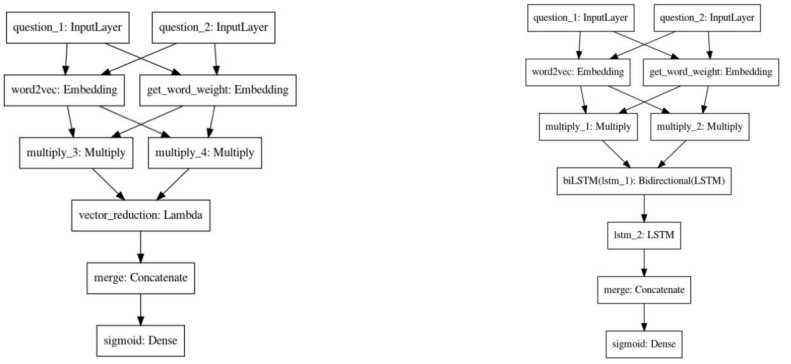
4 Methods

4.1 Siamese Networks

We started with a simple model. In this simple model, we first took the average of the pre-trained word vectors in each question and then compute the cosine distance between the mean word vectors of each question. We can then built a logistic regression model based on the cosine distance. This simple model gave testing auc of 0.7139 and testing accuracy of 0.6656 (Cf. Table 1 word2vec_v1). We further made the model more sophisticated by manually adding back seven stop words "what","when","where","which","who","whom","how". Those words turned out to be useful for identifying duplicate questions. (Cf. Table 1 wrod2vec_v2). Instead of taking simple average, we also tried to take tf-idf weighted average. This slightly increased the testing auc to 0.7263 and testing accuracy to 0.6762 (Cf. Table 1 word2vec_v3). We then tried some neural network architectures. The *siamese_v1* (Figure 1 (a)) can be viewed of the generalization of the tf-idf weighted word2vec model. Note that if the weights in the embedding layer *get_word_weight* are all ones, then this siamese network is the same as simple averaging mode *word2vec_v1*; if the weights are the tf-idf, then this siamese network is the same as the tf-idf weighted model *word2vec_v3*. This siamese network is hence a generalization of the word2vec model. This model substantially increased the testing auc to 0.7982 and testing accuracy of 0.7470. Then we made the network more sophisticated by replacing the *vector_reduction* layer (which is simply a summation) with various LSTM layers. This leads to even better performance. The results are tabulated in Table 1. The best siamese model we got is with a bidirectional LSTM layer followed by another LSTM layer (Cf. Figure 1 (b)). This model gave testing auc of 0.8261 and testing accuracy of 0.7693. All results are report in section 5.

4.2 Deep bi-LSTM Architectures

We also tried several Deep Architectures starting with a bi-LSTM followed by Fully Connected(FC) layers after the pooling/attention layer (Figure 2). Xavier Initialization for the FC Network and Identity Matrix for the biLSTM. Regularization strategies for the linear layer included dropout, weight decay and batch normalization. Normalization for the RNN is not as straightforward. GPU memory requirements increases linearly with the number of layers in a RNN and naive dropout is only added between layers. Tradeoff is increasing num layers in a RNN requires a reduction in the hidden layers(n_h). Increasing num layers should increase the ability of a deep network to learn higher order features but reducing n_h decreases performance as seen by the table showing a reduction in



(a) (b)

Figure 1: (a): A siamese network which is a generalization of tf-idf weighted word2vec model; (b) A siamese network which replaces *vector_reduction* with a bidirectional LSTM layer followed by another LSTM layer.

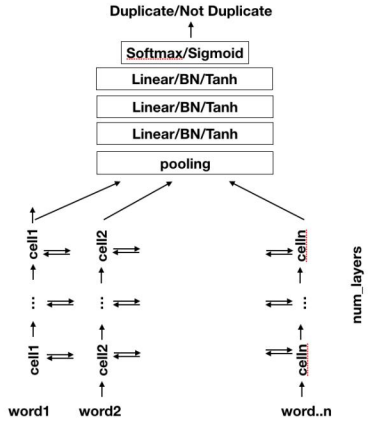


Figure 2: A deep bi-LSTM

test/validation error when going from n_h=4096 to n_h=2048. When optimizing regularization with dropout, 2048 n_h and layer=2 takes 9.3G of single GPU memory which is our limit for a single 12G TitanX. A 16G v100 does not help in being able to fit 2*9.3G. We can see the dropout regularization reduces the difference between training and dev set error from the non regularized architectures. Larger dropout values greater than .2 cause substantial degradation in performance(less than .5 accuracy), more than a traditional CNN, probably because there is no convolution. Hierarchical Attention performs poorly because this dataset is a distribution favoring single sentence documents. The best performing network was with dropout and Inner Attention.

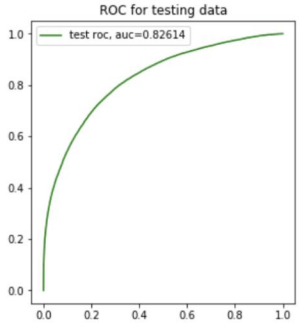
5 Experiments/Results/Discussion

5.1 Siamese Networks

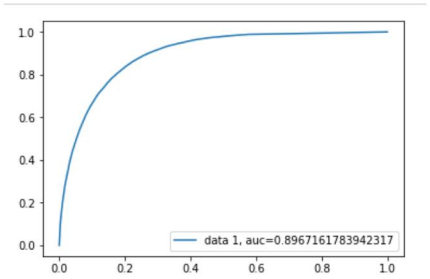
The results of the baseline word2vec models and the various siamese network we tried are in Table 1. The best siamese network we have got consists of an embedding layer for learning the weights for word vectors, and then biLSTM layer followed by another LSTM layer (Figure 1 (b)). The ROC curve for this network is shown in Figure 3 (a). All LSTM layers have 100 hidden units. Regularization is done by early stopping with patience set to 10 epoches and validation ratio set to 0.1. Note that without early stopping, this model can have training auc close to 0.98. With early stopping, the training auc and testing auc is close and that indicates the overfit is mitigated. We also tried adding a dropout layer with keep rate set to 0.8 after the *get_word_weight* layer (this layer alone

Model	auc_train	auc_test	accuracy_train	accuracy_test
word2vec_v1	0.7143	0.7139	0.6668	0.6656
word2vec_v2	0.7236	0.7229	0.6718	0.6714
word2vec_v3	0.7287	0.7263	0.6790	0.6762
siamese_v1	0.8286	0.7982	0.7632	0.7470
siamese LSTM	0.8578	0.8235	0.7906	0.7669
siamese LSTM(2)	0.8642	0.8252	0.7957	0.7687
siamese biLSTM+LSTM	0.8630	0.8261	0.7969	0.7693

Table 1: Results for siamese networks vs. simple word2vec models



(a) ROC for siamese biLSTM+LSTM (Figure. 1(b))



(b) ROC for a deep bi-LSTM (Figure. 2)

Figure 3

has 400k trainable parameters while the total model has 681k trainable parameters). This, however, was not effective in mitigating the overfit. Dropout layers probably would work but requires more tuning. The batch_size in this experiment was set to 8192 which led to the most efficient computation in Telsa V100 16GB GPU.

5.2 Deep bi-LSTM networks

The results for deep bi-LSTM networks are shown in Table 2. Regularization strategies for the linear layer included dropout, weight decay and batch normalization. Normalization for the RNN is not as straightforward. Training time increases linearly with the number of layers in a RNN and naive dropout[13] is added between layers.

6 Conclusion/Future Work

We have tried various network architectures along two directions. The siamese networks aims to generalize the the tf-idf weighted word2vec models. This generalization indeed helps improve the performance greatly. We also tried deep networks aims to adapt the infersent[5] architecture to duplicate detection. For bi-LSTMs regularization of the RNN was the hardest task. The tools we used are of limited effectiveness. Merity[13] has a QRNN but it is not bidirectional. Future work

Model	accuracy_train	accuracy_test	accuracy_diff
Bi-LSTM linear	62.8	62.7	.3
Bi-LSTM(I) 3L	89.0	78.7	10.4
Bi-LSTM(I) 3L Dropout n_h=2048 n_l=2	83.3	78.8	4.5
Bi-LSTM(I) 3L Relu n_h=4096	90.2	79.0	11.2
Bi-LSTM(IA,$numlayer = 2$) 3L Tanh $n_h = 2048$	95.06	83.4	11.65
Bi-LSTM(mean pool) 3L Tanh $n_h = 2048$	94.53	79.31	15.2
hierarchial attention	76.9	77.8	.59

Table 2: Results for deep bi-LSTM networks

would involve testing a bidirectional AWS-LSTM or QRNN. The dataset was created manually using cheap labor in India. A more relevant task would be to create a domain specific dataset perhaps using the Quora dataset or SNLI as a starting point for transfer learning. Most research papers cite SNLI as a task but it is centered around Entailment vs. Duplicate question detection.

7 Contributions

Shiyuan Gu implemented Siamese Networks: https://github.com/shiyuangu/cs230-project
Doug Chang implemented Deep bi-LSTM Architectures: https://github.com/dougc333/cs230_project

References

[1] Yang Liu, Chengjie Sun, Xiaolong Wang Learning Natural Language Inference using Bidirectional LSTM model and Inner Attention, Harbin Institute of Technology, arXiv:1605:09090v1 2016

[2] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, Yoshua Bengio A Structured Self-Attentive Sentence Embedding, IBM Watson Montreal Institute for Learning Algoritmhs(MILA), Universite de Montreal, arXiv:1703.03130v1 2017

[3] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, Eduard Hovy (2018) Hierarchical Attention Networks for Document Classification, Carnegie Mellon University, Microsoft Research Redmond ??? arXiv:XXXXX 2017

[4] Bingning Wang, Kang Liu, Jun Zhao, Inner Attention based Recurrent Neural Networks for Answer Selection, National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing China , Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics pages 1288-1297 Berlin, Germany August 7-12 2016 Association for Computational Linguistics

[5] Alexis Conneau, Douwe Kieal, Holger Schwenk, Luc Barrault, Antone Bordes Supervised Learning of Universal Sentence Representations from Natural Language Inference Data, EMNLP arXiv:1705.02364v5 2018

[6] Infersent code: https://github.com/facebookresearch/InferSent

[7] Hierarchial Attention Networks for Document Classification:
https://github.com/richliao/texthttps://www.overleaf.com/project/5c102586be8c192445108f11Classifier

[8] A Structured Self-Attentive Sentence Embedding: https://github.com/ExplorerFreda/Structured-Self-Attentive-Sentence-Embedding

[9] Pytorch: https://pytorch.org/

[10] scikit-learn:https://scikit-learn.org/

[11] spaCy: https://spacy.io/

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

[13] Stephen Merity, Nitish Shirish Keskar, Richard Socher, Regularizing and Optimizing LSTM Language Models 2017 arXiv:1708.02182v1

[14] A theoretically grounded Application of Dropout to Recurrent Neural Networks

[15] Quora Dataset for duplicate question, https://www.kaggle.com/sambit7/first-quora-dataset

[16] Quoc Le, Thomas Mikolov, Distributed Representations of Sentences and Documents, (2014) axRiv:1405.4053

[17] Travis Addair Duplicate Question Pair Detection with Deep Learning, cs224n project, https://web.stanford.edu/class/cs224n/reports/2759336.pdf

[18] Pretrained word vectors: https://nlp.stanford.edu/projects/glove/