Analysis of Semantic Similarity Methods Applied to the Problem of Duplicate Question Detection

Rafe Kruse
Herbert Wertheim College of Engineering
University of Florida
rafekruse@ufl.edu

Abstract

In this paper I explore and analyze text feature extraction methods, logistical regression models, and deep neural network models related to the problem of duplicate question detection. Both these models were applied to the popular dataset Quora Question Pairs. For this problem, the logistic regression model made use of TF-IDF and SVD for text transformation and predicted duplicates by minimizing loss using a stochastic average gradient algorithm. Experimentation showed that a combination of multiple feature creation techniques to create a aggregate feature set resulted in the greatest accuracy. This result demonstrated the importance of having a large set of representative features, however, the logistic model was unable to out preform the novel DNN approach that relied upon a LSTM and convolutional architecture to obtain near state-of-the-art accuracy results.

1 Introduction

Semantic similarity is a subsection of natural language processing (NLP) with the goal of comparing two independent texts to find the degree that the text content is related. Duplicate question detection (DQD) is a large application of semantic similarity algorithms, and the problem I will be analyzing. DQD algorithms are currently some of the primary systems driving many large peer-to-peer question and answer platforms such as Stack Overflow and Quora.

The problem of DQD is especially interesting as it requires grasping the semantic meaning of two pieces of text accurately. When comparing two questions syntactical, metrics such as word count are often uninformative. This means understanding the underlying meaning of the text in the questions is fundamental to solving DQD problems. Once a semantic representation of the text is created it is then important for successful DQD to determine if those two underlying interpretations of the text are correlated to a high enough degree to be considered duplicates. The problem of DQD can be broken down into two fundamental steps that are common among many machine learning problems. First informative feature selection, in the case of DQD, where meaningful, quantifiable metrics of the underlying semantics of the text are created. Second the application of algorithms to quantify the relationship of the two feature sets.

With such a large variety of algorithms capable of operating on NLP features, solving the problem of DQD has been attempted in many different ways. The question answer site Quora has previously used a combination of hand obtained features and random forests [4] for their automated DQD systems to great success. Similar techniques dependent upon combining a large number of semantic features with traditional regression [3] or tree-based models [1] have also shown near state-of-the-art accuracy. Alternatively, with the growing popularity of neural networks, deep learning systems have shown promise in providing accurate DQD solutions that don't rely on a large number of difficult to create features. Recently convolutional neural networks (CNNs) and long short-term memory networks (LSTMs) have been shown to have comparable results [2] to more finely crafted methods.

The Quora Question Pairs is a dataset initially released as a part of a Kaggle competition. As a result of this the data is well formatted and prelabeled making it ideal for training for DQD. During my testing I applied three well known feature transformation techniques, tf-idf, SVD, and GLoVe[5]. Then a traditional logistic regression algorithm was applied to various sets of features. These results served as a baseline comparison to a novel deep learning approach[11] on the same dataset.

2 Problem Statement

2.1 Data

The chosen dataset Quora Question Pairs consists of 400,000 question pairs from Quora's public platform. Each pair of questions contains the following,

id	qid1	qid2	question1	question2	is_duplicate
0	1	2	What is the step by step guide to invest in sh...	What is the step by step guide to invest in sh...	0
1	3	4	What is the story of Kohinoor (Koh-i-Noor) Dia...	What would happen if the Indian government sto...	0
2	5	6	How can I increase the speed of my internet co...	How can Internet speed be increased by hacking...	0

Figure 1: Quora Question Pairs example data.

The question dataset has a negative skew with ≈63% of all pairs be labelled as non-duplicates. Additionally, approximately ≈80% of questions are repeated at least once in different unique question pairs.

2.2 Similarity Metric Function

The underlying function for DQD algorithms in this paper is a distance or similarity function between two question vectors. After the questions are converted into their respective sentence vectors a similarity function gives a quantifiable degree of correlation between vectors. This function takes the general form,

$$\mathcal{S}(q_1, q_2) \tag{1}$$

where q_1 and q_2 are the vector representation of questions one and two. This similarity function can take many vector distances forms[9], however, during testing the cosine vector distance function resulted in the greatest accuracy.

This similarity function can then be applied to various prediction methods to obtain the probability of a question pair being a duplicate. The first method I applied was traditional logistic regression[10] as follows,

$$\hat{y} = \log(\alpha * \mathcal{S}(q_1, q_2) + \beta), \tag{2}$$

where α, $\beta \in \mathbb{R}$ are model parameters.

The deep learning implementation [11] doesn't explicitly define the similarity function but instead makes use of a neural network to approximate the function. The full neural network duplicate prediction function takes the form,

$$\hat{y} = \sigma(\eta(q_1, q_2)), \tag{3}$$

where σ is the final sigmoid activation in the full neural network. η represents the multilayered network consisting of a series of merge, convolution1D, dense, PreLU, dropout, and batch normalization layers(Reference).

2.3 Loss Functions

For the loss function of the logistic regression problem, a stochastic average gradient algorithm, SAG[6] in particular was chosen to maximize accuracy. In order to minimize overfitting and extreme

weighting of certain words in the questions a l_2 regularization term was added to the loss function. This results in the following total loss function,

$$\underset{x\in\mathbb{R}}{\text{minimize}}\frac{1}{n}\sum_{i=1}^{n} log(1+exp(-b_i\alpha_i^T x)) + \frac{\lambda}{2}||x||^2 \tag{4}$$

where b, α are function parameters and λ follows the default sklearn[7] SAG parameter value of $\lambda = \frac{1}{n}$.

For the deep learning model the loss function is the Keras[8] libraries implementation of the binary cross entropy loss function,

$$L(y,\hat{y}) = -\frac{1}{N}\sum_{i=0}^{N}(y * log(\hat{y}) + (1-y) * log(1-\hat{y})) \tag{5}$$

3 Algorithms

3.1 Feature Extraction

TF-IDF is a well known algorithm commonly used in NLP for vectorizing text and assigning a weighting schemes to a text corpus. TF-IDF works by take the product of the frequency of a term in a document, normalized by document length, and the inverse frequency of the term in all documents. This emphasizes common terms in a document while penalizing terms occurring frequently across all documents, as a result capturing terms relevant to the individual document. The TF-IDF score for term t in document d from the entire document set D can be calculated as follows,

$$tfidf(t,d,D) = tf(t,d) * idf(t,D) \tag{6}$$

$$tf(t,d) = 0.5 + 0.5 * \frac{f(t,d)}{max\{f(t',d) : t' \in d\}} \tag{7}$$

$$idf(t,D) = log(\frac{|D|}{1+|\{d \in D : t \in d\}|}), \tag{8}$$

where,
$f(t,d)$: frequency metric of term t in document d, in the sklearn[7] library, the word count of term t over the total term count of document d.
$1+|\{d \in D : t \in d\}|$: the number of documents where the term t appears, normalized to avoid division by zero.

SVD is a similar algorithm to PCA, with the unique difference that SVD works directly on the data matrix as apposed to the covariance matrix. SVD serves to extract the k principal components of the data matrix in order to reduce dimensionality.

$$X \approx X_k = U_k\Sigma_k V_k^T \tag{9}$$

where $U_k\Sigma_k^T$ is the new decomposed data with k features for training.

GLoVe short for Global Vectors for Word Representation is an unsupervised learning algorithm that extracts term vector representations from a unified corpus. The GLoVe algorithm is trained on word-word co-occurence within the chosen corpus. Because the GLoVe algorithm gains a better understanding of text when given a very large corpus, and can generalize across applications, I chose to use the popular pretrained 840B.300d vector set. The 840B.300d vector set was initially trained on 840B tokens and 2.2M cased terms resulting in 300 dimensional term vectors.

3.2 Model

Logistic Regression was the primary algorithm I implemented during testing as a comparison to a DNN based approach. The sklearn library was utilized to run logistic regression using the SAG solver for fitting. From the available optimization methods, SAG resulted in the highest accuracy on the Quora Question Pair dataset as can be seen in section 4.2. The the SAG algorithm[6] is a variant of traditional stochastic gradient methods with the benefit of a iteration cost that is independent of

the number of terms in the sum. Additionally SAG includes a gradient memory that increases the algorithms convergence rate. The programmatic implementation for minimizing of the SAG loss function in section 2.3 is shown below,

Algorithm 1: SAG minimization with l_2 regularization, with step size α and y prediction targets

$n, f = x.shape[0], x.shape[1]$
$w = 0$ for $i = 1, 2, ..., f$
$s = 0$ for $i = 1, 2, ..., f$
$M = 0$ for $i = 1, 2, ..., n$
$c = 0$
for *k = 0, 1,...,i* **do**
  for *i = 0, 1,...,n* **do**
    Sample j from 1,2,...,n
    if *j hasn't been sampled* **then**
      $c = c + 1$
    end
    $p = jw$
    $g = log(p * y)$
    $u = jg + \alpha w$
    $s = u - M_j \; M_j = u \; w = w - \alpha * \frac{s}{|c|}$
  end
end

DNNs have become increasing popular for their ability to generalize to different problems. The model implementation[11] that I include in this paper as a deep learning baseline was specifically designed for the Quora Question Pairs dataset during the Kaggle competition. This model was implemented using the Tensorflow and Keras libraries and applies the popular Adam[12] algorithm for stochastic optimization. The reference algorithm makes use of the GLoVe embeddings as the feature set and relies upon a merge of both convolutional and lstm sub-networks into a single feed-forward network.

4 Experiments

4.1 Data Processing

The Quora Question Pairs dataset was initially well formatted, however, a few steps had to be taken to remove minor inconsistencies in the data.

1. Remove id, qid1, qid2 data from the question pairs
2. Remove pairs with empty questions
3. remove invalid character/words.
4. Remove stop words from the question set

From this cleaned data five unique vectorizations were created,

- x_1 : question1 and question2 are converted with TF-IDF seperately then stacked feature-wise.
- x_2 : question1 and question2 are stacked feature-wise and converted with TF-IDF together.
- x_3 : Each question1 and question2 are converted with TF-IDF seperately then ran through SVD after which the resulting matrices are stacked feature-wise.
- x_c : Datasets x_1, x_2, and x_3 are all stacked feature-wise.
- x_g : question1, question2, and is_duplicate are converted using the pretrained 840B.300d GLoVe word embeddings.

4.2 Model Parameters

As previously mentioned the primary algorithm comparison was of logistic regression applied to different feature sets and novel deep learning technique with near state-of-the-art accuracy. Logistic regression was implemented using the sklearn library and parameter tuned to maximize accuracy. The four primary parameters with the greatest impact on accuracy, are the similarity function, regularization type, regression solver, and n_gram degree. The accuracies for the different parameters can be found in tables 1-4 where the accuracy shown is the highest value with the respective variable held constant while all other parameters were maximized.

Table 1-4: hyperparameter accuracy comparison on ideal datasets.

Similarity Function	Accuracy
Cosine	**0.778**
Manhatten	0.574
Mahalanobis	0.628
Euclidean	0.562

n_gram	Accuracy
unigrams	0.739
bigrams	0.648
trigrams	0.614
all	**0.778**

Regression Solver	Accuracy
SAGA	0.742
SAG	**0.778**
liblinear	0.648
newton-cg	0.673

Regularization	Accuracy
l_1	0.724
$\boldsymbol{l_2}$	**0.778**
elastic	0.760
none	0.659

4.3 Results

The final ideal logistic regression model was run on each of the datasets with a 90/10 training test split and random shuffle sampling. The spread of logistic regression accuracies demonstrates the importance of feature set selection when training any specific model.

Table 5: Feature set with ideal hyperparameters accuracy comparison

Feature Set	Accuracy
x_1	0.761
x_2	0.760
x_3	0.679
x_c	0.778
x_g	0.764
DNN	0.812

*DNN accuracy was for model after 40 epochs of training, the original source of the algorithm obtained an accuracy of 0.848 after ≈150 epochs

The logistic regression results show that having the combined feature set out preformed all of feature sets. This indicates the importance of having a large feature spread allowing for more generalizations. The combination of feature sets likely had a similar impact to model averaging techniques, namely ensemble learning, where the multi-model average outperforms any individual. With a rather small amount of tuning, logistic regression was able to obtain an accuracy of 0.778.

5 Conclusion

In this paper I investigated how hyperparameters and feature set construction can effect the final accuracy results of a DQD problem. It is clear from experiments that regardless of the chosen model, the methods for feature extraction and processing of data are paramount to a successful result.

Additionally from the results, the DNN model expectedly outpreformed the more traditional logistic regression model, however, the regression model demonstrated that less compute intensive NLP

methods can still be effective given proper parameter tuning. This is likely the reason, despite the success of deep learning, Quora still employs a highly tuned random forest model[4] for the problem of DQD.

In future work I would be interested to see if DNN based models could be tuned to a degree comparable to the state-of-the-art random forest model despite their enigmatic nature. Given that neural networks tend to benefit from being supplied increasing amounts of data It would be informative if a DNN model could obtain a superior results when trained on the entire Quora question database. Lastly and arguable the most important metric for the methods explored in this paper is if they could gain comparable results on similar datasets with minimal retraining.

References

[1] Sharma, Lakshay & Graesser, Laura & Nangia, Nikita & Evci, Utku. (2019). Natural Language Understanding with the Quora Question Pairs Dataset. https://arxiv.org/ftp/arxiv/papers/1907/1907.01041.pdf

[2] Addair, T.G. (2017). Duplicate Question Pair Detection with Deep Learning. https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1174/reports/2759336.pdf

[3] Bär, D., Biemann, C., Gurevych, I., & Zesch, T. (2012). UKP: Computing Semantic Textual Similarity by Combining Multiple Content Similarity Measures. SemEval@NAACL-HLT. https://www.aclweb.org/anthology/S12-1059.pdf

[4] Nikhil Dandekar. 2017. Semantic question matching with deep learning. "https://engineering.quora.com/Semantic-QuestionMatching-with-Deep-Learning".

[5] Pennington, Jeffrey & Socher, Richard & Manning, Christoper. (2014). Glove: Global Vectors for Word Representation. EMNLP. 14. 1532-1543. 10.3115/v1/D14-1162. https://www.aclweb.org/anthology/D14-1162.pdf

[6] Mark Schmidt, Nicolas Le Roux, Francis Bach. Minimizing Finite Sums with the Stochastic Average Gradient. Mathematical Programming B, Springer, 2017, 162 (1-2), pp.83-112. ffhal-00860051v2f

[7] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine Learning in Python. J. Mach. Learn. Res. 12, null (November 2011), 2825–2830.

[8] Chollet, F., & others. (2015). Keras. https://keras.io.

[9] Shah, P. (2018). Adversarial Domain Adaptation for Duplicate Question Detection. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (pp. 1056–1063). Association for Computational Linguistics.

[10] Yoshi Homma, Stuart Sy, and Christopher Yeh. 2016. Detecting Duplicate Questions with Deep Learning. In Proceedings of the International Conference on Neural Information Processing Systems (NIPS.

[11] Abhishek Thakur. (2017). Is That a Duplicate Quora Questions?. https://github.com/abhishekkrthakur/is_that_a_duplicate_quora_question

[12] Kingma, Diederik Ba, Jimmy. (2014). Adam: A Method for Stochastic Optimization. International Conference on Learning Representations.