Deducing Similarity from Quora Question Pairs

Esther LING
lingesther@gatech.edu

LIU Conghai
lconghai3@gatech.edu

Melvin MATHEW
melvin.julian.mathew@gatech.edu

Summary

Quora's question pairs challenge is essentially a problem of quantifying semantic similarity between two short texts. To do this, we generate sentence embeddings of each question using pre-trained word vectors. We apply a distance measurement on the two question vectors and combine them with handcrafted features in a random forest model. We also train an LSTM neural network. Testing on Kaggle's 2 million unlabelled dataset:, the random forest model yielded a log-loss of 0.43518, with rank 955/1588 teams, while LSTM gave our best score of LB 0.29832, ranking 185/1588 teams.

Feature Engineering (Lead: Esther)

We experiment with different sentence embeddings:

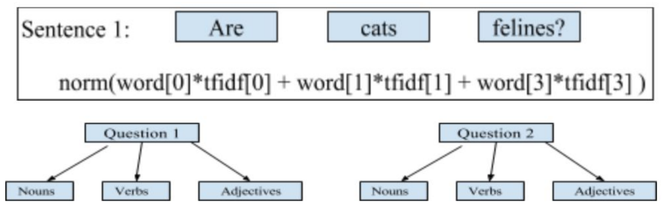
- Apply tf-idf weights to each word vector
- Create new vectors from parts-of-speech (POS) and Named-Entity-Recognizer (NER) tags based on the count (POSNER vectors)
- Augment POSNER vectors to tf-idf weighted word vectors
- Augment POSNER tf-idf weighted word vectors with basic features

WordNet Path Similarity:

- Create sub-sentences (nouns, verbs, adjectives) and measure path similarity (noun-noun;verb-verb;adj-adj)

Basic features:

- Number of words, characters, common words
- Levenshtein distance between sentences



Feature Evaluation (Lead: Melvin)

Apply a wrapper method (Recursive Feature Elimination). The best subset of features (of size 5) after running feature evaluation on all 65 features:

- Euclidean Distance (word2vec)
- Canberra Distance (word2vec)
- Cosine Distance (tf-idf + POSNER + basic features word2vec)
- Cosine Distance (tfidf word2vec)
- Minkowski Distance (word2vec)

Classification (Lead: Melvin)

Trained a random forest model using 251 trees.

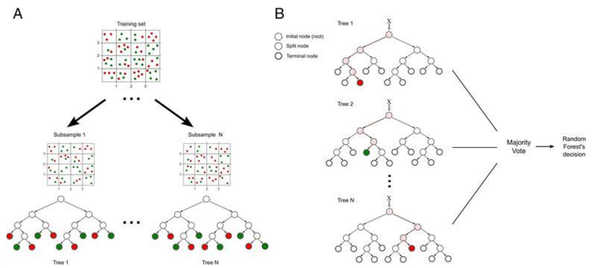
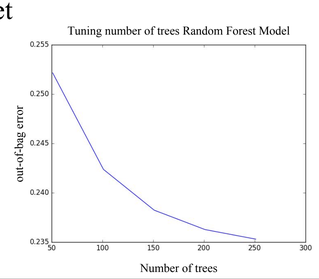


Image source:
https://www.researchgate.net/figure/280533599_fig5_Figure-2-Random-forest-model-Example-of-training-and-classification-processes-using

The training procedure is as follows:

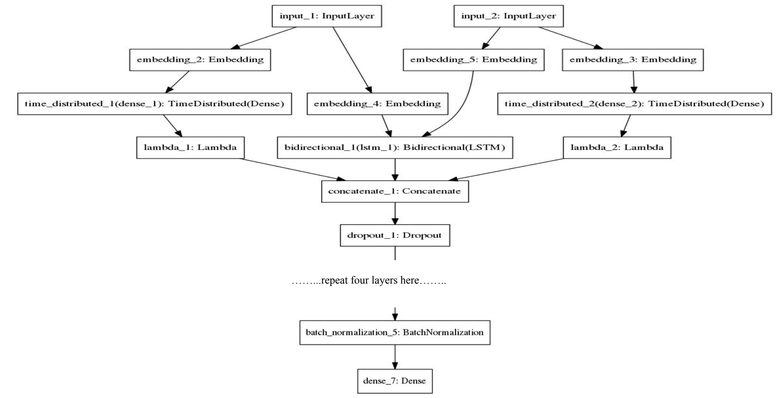
- Split dataset features; 80% training, 20% testing
- Balance training set
- Normalize the balanced training set
- Tune parameters; Cross-validation + grid search
- Train model on full training set
- Normalize the testing set
- Test classifier on testing set

The figure on the right shows the convergence of tuning the number of trees for the random forest model.



Neural Network (Lead: Conghai)

We implement a Glove/Word2Vec+LSTM model on Keras. We tried both Stanford Natural Language Inference benchmark and Google News Word2vec tool, specifically use those Glove word embedding to represent each question in the pair.



Results

Classifier	Training Time (approx)	Original Dataset [Test] Log-Loss	Kaggle Test Set Log-Loss
Logistic Regression [imbalanced]	1 minute	0.27252 (70.22% accuracy)	N/A
Logistic Regression [balanced]	30 seconds	0.1897 (69.49% accuracy)	N/A
Random Forest Model [251 trees]	5-15 minutes	0.22352 (76.72% accuracy)	0.43518 (Rank 955/1588)
Neural Network	1-2 hours	0.2561 (86% accuracy)	0.29832 (Rank 185/1588)

Conclusions

- Major difficulty still lies in finding good embeddings for short text fragments
- Still room for research to develop better sentence embeddings
- Given the same embedding, neural network was able to approximate the better model
- Suggests that the better approach is to develop sentence embedding and feed it directly to neural network