Towards Automatic Evaluation of Presentation Entailment with Original Papers Through Textual Entailment Models

Joshua Segal
Northeastern University
Boston, MA
segal.jo@northeastern.edu

Cristhofer Hernandez
Northeastern University
Boston, MA
hernandez.cri@northeastern.edu

Brendan Brady
Northeastern University
Boston, MA
brady.br@northeastern.edu

Abstract

This paper proposes an automated approach to assess the coherence and similarity between presentation slides and research papers using textual entailment models. Three machine learning models: LSTM RNN, Transformer, and Logistic Regression, are employed to evaluate presentation-paper pairs. The study utilizes the SNLI corpus and ASGSP dataset and assesses model performance using precision, recall, F1-score, and accuracy metrics. While logistic regression exhibited the best performance, the findings underscore the importance of model complexity and fine-tuning. This research sheds light on the potential of using textual entailment models for evaluating presentation coherence and suggests avenues for future exploration.

1. Introduction

Understanding the relationship between pairs of sentences, known as Recognizing Textual Entailment (RTE) or textual entailment, is a subclass of Natural Language Processing (NLP) and a part of the basis of Natural Language Understanding (NLU). Textual entailment follows the relationship between two texts, the *hypothesis* and the *premise*. A textual premise, *P*, entails a textual hypothesis*, H,* if *H* is considered true regarding *P*. For example: consider the sentence "a dog runs across the grass." as a premise, and the sentence "The dog is outside." as a hypothesis. We see that the hypothesis statement can be inferred from the premise, this means there is an entailment textual relationship between the sentences. Consider another *H*: "The cat is thirsty." This cannot be inferred from *P* and therefore has a neutral textual relationship. Consider one last example: "The dog is sleeping." This is a contradictory inference from *P* and therefore has a contradictory textual relationship.

Relationship	Premise & Hypothesis	
Entailment	Premise:	This church choir sings to the masses as they sing joyous songs from the book at a church.
	Hypothesis:	The church is filled with song.
Neutral	Premise:	This church choir sings to the masses as they sing joyous songs from the book at a church.
	Hypothesis:	The church has cracks in the ceiling.
Contradict	Premise:	This church choir sings to the masses as they sing joyous songs from the book at a church.
	Hypothesis:	A choir singing at a baseball game.

We begin this report by discussing a novel application of the textual entailment task, evaluating presentation and research paper pairs on entailment for similarity and cohesiveness. Next we discuss the datasets and models used for this task, and finally our results, discussion, and future work.

1.1 Objectives of the Research

The objective of this research is to present three different machine learning models' effectiveness at approaching the natural language processing (NLP) task of text entailment in order to evaluate presentation's levels of similarity with a wide range of research papers.

1.2 Significance of the Study

Making a presentation is a very important skill throughout all fields of academia and industry. According to a poll by Poll Everywhere, over 35 million presentations are given every day to over 500 million audiences (Everywhere). A common difficulty of creating a presentation is knowing what to include and how to include it. We aim to alleviate some of the qualitative consideration of how "good" a presentation is with quantitative metrics that can serve as a guideline for presentation creation and fine tuning.

2. Data

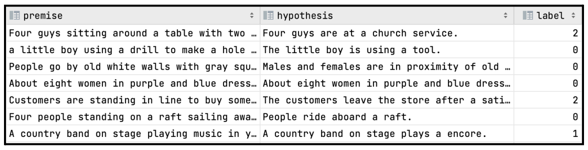
This section aims to provide a comprehensive understanding of the datasets utilized for training text entailment models and evaluating presentation-paper pairs. By characterizing the data, we can assess its suitability for the respective tasks and ensure the accuracy and reliability of our research findings.

Two datasets were utilized across this project. The first dataset used is the Stanford Natural Language Inference (SNLI) corpus, comprising human-written English sentence pairs. The second dataset used is the Automatic Slide Generation from Scientific Papers (ASGSP) dataset consisting of paper-slide pairs.

2.1 Textual Entailment Dataset

The SNLI dataset consists of 570,000 sentence pairs, with 550,000 pairs allocated for training and 10,000 pairs each for development and testing. The samples contain varying degrees of linguistic complexity for generalized training.

The structure is two columns of textual data for the premise and hypothesis sentence pairs, and a third column for the integer label given to the pair. 0 for the entailment class, 1 for the neutral class, and 2 for the contradiction class.



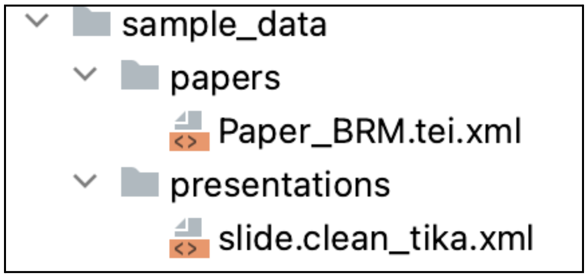
premise	hypothesis	label
Four guys sitting around a table with two …	Four guys are at a church service.	2
a little boy using a drill to make a hole …	The little boy is using a tool.	0
People go by old white walls with gray squ…	Males and females are in proximity of old …	0
About eight women in purple and blue dress…	About eight women in purple and blue dress…	0
Customers are standing in line to buy some…	The customers leave the store after a sati…	2
Four people standing on a raft sailing awa…	People ride aboard a raft.	0
A country band on stage playing music in y…	A country band on stage plays a encore.	1

The SNLI dataset was sourced through Kaggle.com, an online community of data scientists and machine learning practitioners in which many datasets are publicly available. The authors obtained the data through crowdsourcing marketplace platform Amazon Mechanical Turk–a marketplace for completion of virtual tasks that requires human intelligence–in which hired workers were tasked with creating hypothesis sentences of each label (entailment, neutral, contradiction) based on provided premises. Cross-validation was performed by the authors to ensure correctness of generated sentence pairs. Since the authors of the dataset maintained balanced samples across training, validation, and testing data, no balancing practices were employed by us.

2.2 Presentation-Paper Dataset

The ASGSP dataset consists of 5,000 paper-presentation pairs indexed matched for presentations created from the content in the corresponding research paper. The dataset is domain specific to scientific reports and thus has an increased linguistic complexity.

The dataset is unlabeled, and the original structure was that of 5000 subfolders containing XML and pdf files for both the research paper and matching presentation. In order to load the data sequentially by indexes, we refactored the structure to consist of two folders, presentations and papers, and loaded each pair into their respective folder while maintaining index ordering. We also had to clean up a handful of bad data samples that contained no XML files or incomplete XML representations of the pdfs.


```
sample_data
  papers
    Paper_BRM.tei.xml
  presentations
    slide.clean_tika.xml
```

During our human evaluation section, we will discuss more the process of manually labeling combinations of paper-presentation pairs for qualitative similarity of topics/keywords for human benchmarking of our models.

The ASGSP dataset was also sourced through Kaggle.com, and the authors compiled the data from conference proceedings websites.

3. Models

In this section, we present a comprehensive overview of the machine learning models utilized in our research and their evaluation in the context of presentation-paper pairs. By analyzing the characteristics of each model, we seek to evaluate their efficacy and suitability for the tasks at hand, thereby ensuring the accuracy and reliability of our research findings.

Three models were constructed across this project. The first is an LSTM RNN model. The second is a Logistic Regression model. The third is a BERT model.

3.1 LSTM RNN

3.1.1 Selection

We first chose an LSTM RNN model as they excel at capturing long-range dependencies in sequential data, making them well-suited for text entailment tasks. Unlike traditional neural networks, LSTM cells incorporate feedback connections, enabling them to process entire sequences of data rather than individual data points. This capability allows LSTM models to understand and predict patterns in sequential data, such as our textual entailment task, very well.

3.1.2 Preprocessing

In preparing the data for training the LSTM RNN model, several preprocessing steps were involved. The data underwent cleaning, including lowercasing, punctuation removal, and stop-word removal with the stopwords corpus from the NLTK library.

After cleaning, the data was transformed into word embeddings using a word2vec model from the Gensim library. Word embeddings like Word2Vec encode semantic information about words in a dense vector space. This helps capture relationships and meanings between words, which is crucial for NLP tasks. The dense vector space also reduces the dimensionality of the input space, making it computationally more efficient while retaining important semantic information.

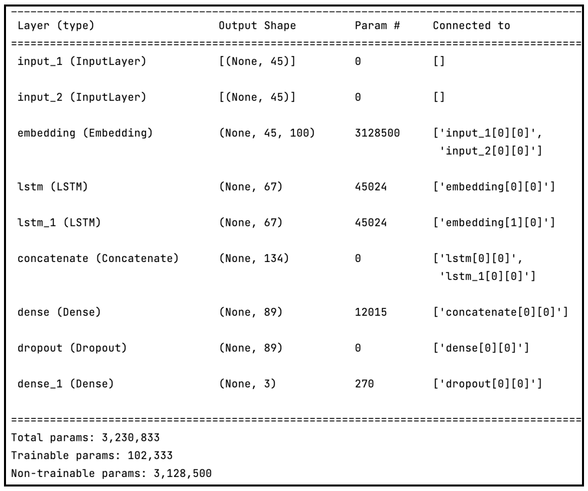
Next, we dynamically padded all sequences to the maximum sequence length with the pad_sequences function from the Keras library. Dynamic padding ensures that all sequences have the same length, which is essential for efficient batch processing during training. Furthermore, padding allows us to accommodate varying lengths of natural text, and mitigate the sequence lengths impact on the model through a standardization process without loss of information through other methods like truncating.

Finally, the labels of the data were one-hot encoded into categorical labels with the to_categorical function from the Keras library. This was done to transform the labels into a format that neural networks can easily process.

The data was split into 80% training data, 10% validation data, and 10% testing data using the train_test_split function from the sklearn library.

3.1.3 Architecture

The architecture of the LSTM model consists of an untrainable embedding layer, loaded with the shape and weights of the Word2Vec model. The premise and hypothesis sentences input data are then passed through the embedding layer separately in order to capture the individual semantics of each sentence before establishing their relationship. After, the embedded sentences are passed to the LSTM layers with 67 units and a 0.28 dropout, separately again to maintain individuality. This layer captures the long-term dependencies in sequential data, the flagship feature of LSTMs. Next, the embedded data is finally concatenated and passed through a dense layer with 89 units and ReLU activation for feature extraction, followed by the final output layer with 3 units and a softmax activation to classify the data.


```
------------------------------------------------------------------------------------------
 Layer (type)                Output Shape         Param #     Connected to
==========================================================================================
 input_1 (InputLayer)        [(None, 45)]         0           []

 input_2 (InputLayer)        [(None, 45)]         0           []

 embedding (Embedding)       (None, 45, 100)      3128500     ['input_1[0][0]',
                                                               'input_2[0][0]']

 lstm (LSTM)                 (None, 67)           45024       ['embedding[0][0]']

 lstm_1 (LSTM)               (None, 67)           45024       ['embedding[1][0]']

 concatenate (Concatenate)   (None, 134)          0           ['lstm[0][0]',
                                                               'lstm_1[0][0]']

 dense (Dense)               (None, 89)           12015       ['concatenate[0][0]']

 dropout (Dropout)           (None, 89)           0           ['dense[0][0]']

 dense_1 (Dense)             (None, 3)            270         ['dropout[0][0]']

==========================================================================================
Total params: 3,230,833
Trainable params: 102,333
Non-trainable params: 3,128,500
```

3.1.4 Training

A data generator was employed to supply the model with repeated training and validation data. This was done for the efficient training on the large SNLI dataset and to allow the model to continue training for the hyperparameters irregardless of the data length.

To find our optimal hyperparameters, Bayesian optimization was employed. The model was trained with a batch size of 32, over 10 epochs, using a learning rate of approximately 0.001. Categorical cross-entropy loss function and the legacy Adam optimizer were utilized.

3.2 Transformer

3.2.1 Selection

We next chose a transformer model since the self-attention mechanism in the transformer architecture allows the model to focus on different parts of the input sequence and capture dependencies. By computing attention scores for each element based on its relationships with other elements in the sequence, the model can understand the relationships between words in a sentence. Positional encoding is essential in the

Transformer model for handling sequence order, injecting information about the position of each token within a sequence.

We specifically used the 'distilbert-base-uncased' model for our transformer implementation due to its lighter and more efficient training speeds.

3.2.2 Preprocessing

The data underwent similar cleaning to the LSTM RNN. Next, our transformer tokenized the data using the 'distilbert-base-uncased' AutoTokenizer, padding to the maximum length sequence, and converting the data to tensors. Labels were also one-hot encoded using the to_categorical function for the same reasons.

3.2.3 Architecture

The architecture of the Transformer model comprised a sequential keras neural network. It is covered by a preloaded TFDistilBertModel layer, a lambda layer that extracts the classification token from the DistilBERT model layer, a dense layer with 64 units and ReLU activation, a dropout layer with a dropout rate of 0.2, another dense layer with 32 units and ReLU activation, and an output layer with 3 units and softmax activation.

3.2.4 Training

As training transformers with BERT embeddings takes a significant amount of time, training was conducted for only 3 epochs. The Adam optimizer was used with a learning rate of 0.0001, batch size was set to 64, and maximum sequence length was set to 32 (this pads all data inputs to a length of 32).

3.3 Logistic Regression

3.3.1 Selection

We lastly chose a logistic regression model as our baseline implementation. Logistic regression models represent text data as numerical features suitable for classification tasks.

3.3.2 Preprocessing

The logistic regression model involved similar data cleaning to the previous models. The text was then transformed using the CountVectorizer from the sklearn library. Logistic regression with CountVectorizer serves as a simple yet effective baseline model for our text entailment task. It provided a good starting point for exploring more complex models.

3.3.3 Training

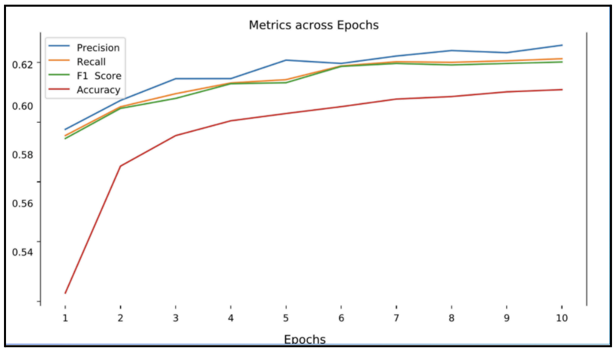
The hyperparameters for the logistic regression model were chosen through a grid search, settling on an L1 penalty, C of 1, and the liblinear solver for optimization. A max_iter of 10,000 was used to ensure convergence.

4. Results and Conclusions

4.1 Main Results

4.1.1 LSTM RNN Performance on Text Entailment Dataset

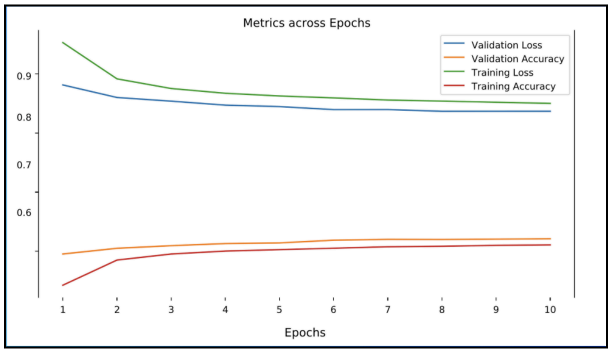
The graph below shows the results of our LSTM RNN model on precision, recall, f-1, and accuracy scores (PRFA), on the validation data across 10 epochs. Our best model was obtained from epoch 9, which had the lowest validation loss of 0.8424 and and validation PRFA scores of 0.624, 0.62, 0.61, and 0.6. These results showed that the model performed effectively across multiple performance metrics, achieving a good balance between precision, recall, and accuracy.



Below is the graph of the same model on validation loss, validation accuracy, training loss, and training accuracy. Through these results we found our best model metrics at:

Loss	Accuracy	Validation Loss	Validation Accuracy
0.857	0.6068	0.8424	0.6165

This demonstrated to us that our model continued to learn throughout all 10 epochs, and did not begin to overfit until the last epoch, signifying a correctly architected model that is learning effectively.



However, our final PRFA scores from the test dataset were unexpected:

	Final
Accuracy	0.332
Precision	0.331
Recall	0.332
F1	0.3309

We believe these unexpected results were due to poor model generalization and dataset shifting through statistical differences on the training and validation datasets compared to the testing dataset.

4.1.2 Transformer Performance on Text Entailment Dataset

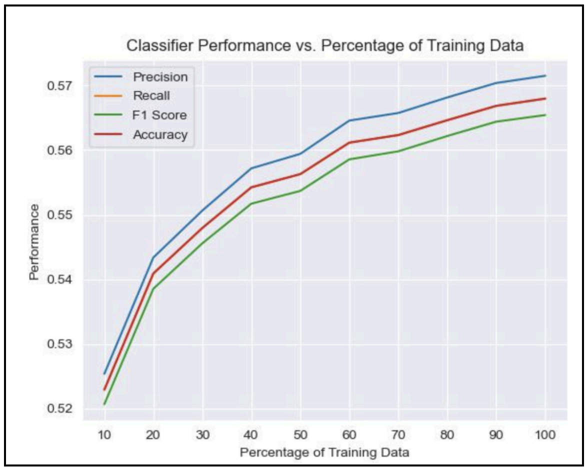
The table below shows the results of our transformer model on precision, recall, f-1, and accuracy scores (PRFA), on the validation data across 3 epochs. These results showed that the model learned effectively across multiple epochs.

	Epoch 1	Epoch 2	Epoch 3
Accuracy	0.3689	0.4355	0.4163
Precision	0.1595	0.1745	0.3724
Recall	0.0032	0.0039	0.0078
F1	0.0062	0.0076	0.0152

We were surprised by how low the recall and f-1 scores were and believe it is due to finetuning and model complexity poorly generalizing to positive data.

4.1.3 Logistic Regression Performance on Text Entailment Dataset

The graph below shows the results of our logistic regression model on precision, recall, f-1, and accuracy scores (PRFA), on the validation data across 10% increments of our data. Our model continued to learn effectively through each incorporation of more data.



Our final epoch 10 PRFA scores show that the model performed effectively across multiple performance metrics, achieving a good balance between precision, recall, and accuracy.

Accuracy	Precision	Recall	F1-Score
0.5261	0.5299	0.5210	0.5244

4.2 Human Evaluation on paper-slide pairs

The overarching objective of this section is to establish a direct relationship between entailment scores calculated by our models and human evaluation scores of presentation entailment. By correlating these metrics, we aim to validate the utility of text entailment in evaluating presentations against research papers accurately.

Model predictions for presentation-paper pairs were conducted through parsing all the sentences from imputed presentation and paper XML files. Then the sentences were matched for maximum similarity through keyword count cosine similarity. These matched sentences were fed through our trained models to get a predicted label. Finally, all the labels were argmaxed to the class labels of 0, 1, 2 (entailment, neutral, contradiction) and a “paper score” was calculated through a normalized weighted sum of the counts of each label, pairs crossing the threshold of 0.7 were labeled as entailed pairs, pairs under were labeled as unrelated pairs.

These predicted labels were scored against the human benchmark labels derived from “general topic and keyword similarity” as follows:

The LSTM RNN predictions compared to human ground truth labeling:

Accuracy	Precision	Recall	F1-Score
0.6162	0.1951	0.1951	0.1951

Logistic Regression compared to human ground truth labeling:

Accuracy	Precision	Recall	F1-Score
0.372	0.240	0.756	0.364

Interestingly, these results were very different in terms of what metric they performed well on, showing us that the different ways the models learned and featurized the data affected their inference time performance.

4.3 Discussion

It is observable that the logistic regression model performed most favorably, with the highest f1, recall, precision, and accuracy scores. The next best model for classification would be the LSTM RNN model, and the least accurate model would be the transformer model.

It is quite surprising that the simplest model did the best overall, while the most complex model did the worst. As models become more complex, it becomes more difficult to fine-tune them for a specific purpose, which makes complex models more vulnerable to overfitting, underfitting, or training for the incorrect purpose.

4.4 Limitations

It was expected that the BERT transformer model would perform best due to its vast number of embeddings and pre-trained knowledge. However, due to the time limit, it was difficult to fine-tune the pretrained DistilBERT base model for a specific classification purpose. Hyperparameters seem well adjusted, as seen by the Transformer metrics with just about twenty minutes of training, however many hours of training will likely be required before the model can produce its best metrics.

5. Future Work

In the future, we wish to further fine-tune our models for a better performance in predicting the correlation between slides and papers. This would involve more epochs for all models, especially the BERT Transformer model. Further experimentations with word embeddings, learning rates, and featurization (such as KMeans clustering) would have to be experimented with to find the optimal solution.

We would use these improved models to add more features to our application, which would create entailment pairs to increase the nuance in sentence matchings. Word embeddings and keyword matching would likely be used for this purpose.

Furthermore, we would like to dynamically match entailment and thus group sentences by similarity. This would be an incredibly helpful tool to express to the user what areas of their presentation match with the most similar paper.

Finally, we would also like to include a rating summary, which would suggest areas of improvement for the presentation, references of important topics, and vital information missed in the slides.

References

Everywhere, P. (2016, November 29). *10 little-known facts about powerpoint [infographic]*. Poll Everywhere Blog. https://blog.polleverywhere.com/powerpoint-infographic