LING 131A

Prof. Marcus Verhagen

Group members: Xiaoyu Lu, Yonglin Wang

Capturing Text Formality of Online Answers: Analyzing GYAFC with Machine Learning Algorithms

Introduction

In this project, we are examining how different linguistic properties of a sentence correlates with the formality of the project and what machine learning models can best predict the formality of a sentence based on a certain feature. More specifically, we are interested in the performance of three different models, namely logistic regression, Naïve Bayes, and decision tree when using each of the following features or feature group: basic statistics, raw text, n-gram, readability, entity types, and entity length. In addition, we also trained a Long Short-Term Memory (LSTM) model with GloVe embeddings, to see how well deep learning algorithms can predict the formality of a sentence without any help of feature extraction.

Before we illustrate the implementation details, it is important to talk about the emerging importance of formality analysis in NLP and give a brief introduction to the Grammarly's Yahoo Answers Formality Corpus (GYAFC).

A large amount of NLU studies focus on the content of written texts. However, the stylistic choices of the author also carry equally significant impact on the information conveyed to their audience. To illustrate this idea, let us consider the following examples:

1. *Those recommendations were unsolicited and undesirable.*
2. *that's the stupidest suggestion EVER.*

We can observe that, although the literal meaning of the two sentences are the same, the first sentence, being much more formal than its informal counterpart, is highly likely to convey a different message when elicited under the same context. Therefore, to achieve full

understanding of language and real advancements in dialog systems, information extraction, and human computer interaction etc., it is imperative to incorporate the non-literal, as well as the literal, into text analysis.

In light of this, researchers at Grammarly have created Grammarly's Yahoo Answers Formality Corpus, the largest dataset containing a total of 110K informal / formal sentence pairs (See Sudha & Tetreault, 2018). All informal sentences in this corpus come from the domains of *Entertainment & Music* and *Family & Relationships* on Yahoo Answers. Then, the formal rewrite rules were presented to collect the formal version of each version on Amazon Mechanical Turk. Table 1 shows some sample sentence pairs. For the sake of brevity in this project, we will only examine about 11K sentence pairs randomly pre-selected by the creators of GYAFC.

Informal Examples	Formal Rewrite Examples
I've watched it and it is AWESOME!!!!	I viewed it and I believe it is a quality program.
She cant sing for her life!	She is a poor vocalist.
i dont know, but he iss wayyyy hottt	He is very attractive.
MAYBE AT 20 YEARS OLD AND YOUR MARRIED.	This is an example of something someone, who is age 20 and married, would do.

Table 1. Corresponding re-written examples from the GYAFC corpus. The formal sentences are rewritten into informal ones according to certain rewrite rules and vice versa.

There have been multiple lines of research done by linguists and sociolinguists on the linguistic characteristics of text formality. In this project, we decided to pick a few of these characteristics, as listed in Table 2, and use them to train machine learning models to predict the formality of a sentence.

Feature	Details
Basic Statistics	Case: the number of entirely capitalized words; binary indicator for whether sentence is lowercase; binary indicator for whether the first word is capitalized
	Lexical: the number of contractions in the sentence, normalized by length; average word length
	Punctuation: the number of '?', '...', and '!' in the sentence

Raw Text	The raw string of the sentence
N-Gram	The unigrams, bigrams, and trigrams appearing in the sentence
Readability	Length of the sentence, in words and characters; Flesch-Kincaid Grade Level readability score
Entity Types	The entity types (e.g. PERSON, LOCATION) occurring in the sentence
Entity Length	The average length, in characters, of PERSON mentions in the sentence

Table 2. Details of the features and feature group used in training machine learning models. Each feature or feature group was vectorized and fed to a model separately. Though capturing three different linguistic aspects of a sentence, basic statistics is treated as one feature by putting all the corresponding numerical values together.

In terms of the machine learning algorithms, we chose to train the following three types of widely implemented algorithms: logistic regression, Naïve Bayes, and decision tree. To our current knowledge, there has not been any literature specifically comparing the performance of these three algorithms. Therefore, we decided to examine their 5-fold cross-validation accuracies when trained on the same feature or feature groups, in hope to find out what model may seem the most suitable for formality classification tasks.

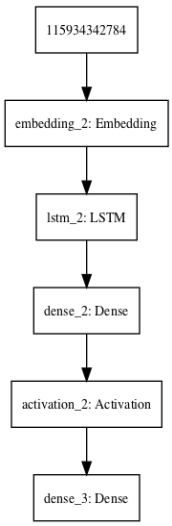

Figure 1. the 5 layers in the deep learning network using LSTM with GloVe in this project. Topmost number is the serial number of our algorithm. Each label down below represents a higher-level layer. In the activation layer activation_2, sigmoid function was used.

In addition, we also looked at the performance of a deep learning algorithm, namely Long Short-Term Memory (LSTM) with Global Vectors (GloVe) embeddings, to determine the effectiveness of deep learning algorithm in classifying text formality. This algorithm uses several layers to progressively extract higher level features from the raw text input. Thus, no feature extraction is required for this algorithm. Figure 1 shows the layers involved in the deep learning neural network. In the Evaluation section, we will compare the performance of this model to other machine learning model trained with features.

In the end, we have decided to try to go beyond analyzing a corpus and made some attempts to predict the formality of a novel input sentence using some of our trained models. We will demonstrate the prediction of a few novel sentences in the Evaluation section of this paper.

Implementation

Feature Extraction

Using separate functions, we extracted the readability feature using readability 0.3.1, subjectivity scores (later ended up not included in the training because the extraction took too long) using TextBlob 0.15.3, and NLTK for the rest of the features in Table 2. For each sentence, we will extract the all the features with specified return data types as shown in Table 3.

Feature	Return Data Type
Basic Statistics	List of numbers
Raw Text	N/A
N-Gram	N/A
Readability	List of numbers
Entity Types	List of unique entity types
Entity Length	List containing a number

Table 3. The return data type of each feature extracted. N/A means the feature was not extracted separately and was instead extracted by the vectorizer.

Feature Encoding (Vectorization)

For each type of feature, we extracted a vector for that feature from each sentence in the entire corpus and saved those vectors in one joblib file. We used Sci-kit learn (sklearn) for encoding these features.

Feature	Vectorizer Used	Feature Array Saved as
Basic Statistics	np.array	fast_num_feature.jbl
Raw Text	LogisticRegression: HashingVectorizer + TfidfTransformer	N/A
	MultinomialNB: CountVectorizer/TfidfTransformer	
	DecisionTreeClassifier: HashingVectorizer/TfidfTransformer	
N-Gram	CountVectorizer(ngram_range=(1,3))	ngram_feature.jbl
Readability	np.array	readability_feature.jbl
Entity Types	CountVectorizer(tokenizer=dummy_processor, preprocessor=dummy_processor, binary=True)	ent_feature.jbl
Entity Length	np.array	entity_length_feature.jbl

Table 4. Vectorizers used for each feature and the file name for the feature arrays. np.array means no vectorizer was used because the data were already numerical; instead, the data were converted into array using the array() function in NumPy. Dummy_processor is a function that returns the data passed to it. The raw text feature was vectorized in pipeline using different set of vectorizers and therefore not saved.

Model Training

Logistic regression, Naïve Bayes, and Decision Tree

Using Sci-kit Learn, we trained three different models, namely, Naive Bayes (MultinomialNB), Logistic Regression (LogisticRegression), and Decision Tree (DecisionTreeClassifier), with each of the features listed in Table 2. For each of the model-feature combination, we split the training set into train and test sets, fit the train set into the model, tested the accuracy using the test set, and did a 5-fold cross-validation with the train set. Note

that, for the raw text feature, pipeline was also implemented. It is easy to control the workflow. In the pipeline, we have vectorizer, transformer and classifier.

After training, each model was saved under *model* folder.

LSTM with GloVe

We also trained a deep learning model with layers described above in Figure 1. We use the pre-trained word vectors from Wikipedia 2014 and Gigaword 5 (this link has a more detailed description for GloVe). The vocabulary size is 400,000. We use the word vectors as weights in the embedding layer. Then we build a Sequential model which contains 5 layers, namely embedding layer, LSTM, Dense, Activation (sigmoid) and Dense (activation='softmax').

After training, the model was also saved under model folder.

Fine tuning

In addition, we fine-tuned all the models trained under raw text features. We use Stratified K fold cross validation and grid search to fine-tune the parameters to find the best parameter.

Prediction of Novel Sentences

In the end, we also attempted to predict the formality of a novel sentence. Each input sentence will be first vectorized based on using the vectorizer described in Table 4 and then passed to a specific model for prediction.

Evaluation

Model Accuracy

Model / Feature	Logistic Regression	Naïve Bayes	Decision Tree	LSTM with GloVe
Basic Statistics	0.808	0.797	0.803	

Raw Text	0.753	0.749	0.500	0.600
N-Gram	0.729	0.687	N/A	
Readability	0.570	0.546	0.598	
Entity Types	0.541	0.540	0.544	
Entity Length	0.526	0.500	0.526	

Table 5. 5-fold cross validation accuracy for each model and feature combination, with LSTM training accuracy displayed on the far right for comparison. Darker shades indicate higher accuracy. Features are ordered based on performance across all models. The decision tree model for N-gram took too long for our computer to train, thus no accuracy is displayed. Note that the models for raw text features have been fine-tuned after training.

Table 5 shows the evaluation results of different types of models we have trained in this project. We can see that logistic regression model has the best overall performance across features. Moreover, it is also suggested that, across models, the basic statistics features can best capture the formality of a sentence, even better than the actual sentence itself. This result does align with our intuition that multiple non-literal characteristics of a sentence together can better predict the formality of the text than the literal content in the sentence.

Prediction

We hereby demonstrate the prediction results of the raw text and LSTM models.

Model / Sentence	Logistic Regression	Naïve Bayes	Decision Tree	LSTM
It was a lovely weekend	formal	formal	informal	formal
if you want a longterm relationship, no way	informal	informal	informal	formal
You love this person in a way your parents dont!	informal	informal	informal	formal
The president frequently injects his own words into statements he claims his supporters said on Fox News or elsewhere."	formal	formal	informal	informal
Mr. T's habit of putting words in the mouths of others are not just limited to impeachment.	formal	formal	informal	formal

Table 6. Labels generated by the models that use only raw text as features. Greyed cells correspond to wrong predictions.

The example sentences of various genres are all selected from online sources. We deem a sentence informal if its spelling is non-standard (e.g. long-term vs longterm) and the word choice is more conversational than formal and writing-like (e.g. impossible vs no way).

We can see that the prediction output in Table 6 corroborates the accuracy shown in Table 5, where the logistic regression and naïve Bayes model correctly predict all sentences due to their high accuracy, while decision tree and LSTM models fall short, getting only two correct labels.

What went wrong

Most notably, the process of feature extraction and model training took unexpectedly long time. Some feature extraction processes took tens of minutes, and model training like decision tree model for n-grams took so long we could not finish the process within half a day.

Moreover, problems arose when we attempted to combine all the features into one vector for each sentence to see if the combination of all features can boost the accuracy. However, due to the extremely large size of the corpus, our computers killed the program every time we tried to stack the feature arrays (exit code 137, interrupted by signal 9: SIGKILL). In the future, we hope to eventually achieve this when the resources allow.

In addition, due to some compatibility issues, there's a portion of models that cannot be loaded for prediction. We are currently working on this but are not positive that the issue can be solved by the deadline.

Work Distribution

Xiaoyu: Model training, Predicting, LSTM training

Yonglin: Corpus loading, feature extraction, feature encoding