Proposition Classification Machine Learning Technical Document

Matt Foulis

April 2020

Contents

1 Executive Summary

This report describes the work carried out in developing a Machine Learning (ML) model with the aim of automatically classifying propositions within text. Using a transcript of a 2016 United States of America Presidential debate, a set of features were developed to train a Support Vector Machine (SVM) model from the scikit-learn framework to classify propositions as either 'Fact', 'Policy', or 'Value'. The model achieved an f_1 score of 0.65. Recommendations for future work are discussed including the creation of new ML features and the gathering and use of further training data to improve the model.

2 Background and motivation

Within an argument, having knowledge of the type of proposition being used can allow further analysis of the argument to be performed. If a proposition describes a factual piece of information, steps can be taken to verify this information. Or, if the proposition proposes some action be taken, consideration of the pros and cons of this can be made.
This work aimed to develop a ML model which could perform this classification automatically. By treating the identification of a proposition's type as a classification task, the model could be trained on labelled proposition data to identify a proposition's type given new data.
For the project, the classifications proposed by Wagemans [2] of 'Fact', 'Policy', and 'Value' are used. A Policy proposition expresses that specific action should be taken, for example, 'We should increase the voting age to 21'. A Value proposition expresses a personal judgement about something, for example 'War and Peace is the best book of all time'. A Fact proposition expresses that something is the case and can be empirically verified, for example, 'Tomorrow will be a Saturday'.

3 Data

For the task, the data used was an annotated transcript of a 2016 United States of America Presidential debate between Hillary Clinton and Donald Trump. Presented by Visser *et al* [1], the dataset contains the propositions from the debate, labelled with the proposition type of 'Fact', 'Policy', 'Value', and 'Blank' for unknown propositions. The dataset contains 796 individual propositions, comprised of: 383 Value, 110 Proposal, 289 Fact, and 14 blank.

4 Implementation

4.1 Frameworks and libraries

A list of the frameworks and libraries used alongside their version numbers can be seen in table 1.

Library/Framework	**Version**
Python	3.7.4
scikit-learn	0.22.1
pandas	0.25.3
Natural Language Toolkit	3.4.5
en-core-web-sm	2.2.5
spaCy	2.2.3
Stanford parser	3.9.2
Speciteller	-*

Table 1: List of the libraries and frameworks used alongside their version numbers

* No version number provided, however the original version, available at: https://github.com/jjessyli/speciteller was used.

4.2 Model

4.2.1 Support Vector Machine

The Support Vector Machine (SVM) model provided by scikit-learn was created using the parameters seen in table 2. These parameters were selected based on the output of a Grid Search over the model hyper-parameters.

Parameter	Value
C	10
kernel	linear
probability	True

Table 2: The parameters and values used for the SVM model

4.3 Features

This section describes the features used to train the models. Features were created and added to a Pandas dataframe. This was then stored in a Comma Seperate Value (CSV) file for use by the model.

4.3.1 Term frequency-inverse document frequency

Using the term frequency-inverse document frequency (TFIDF) Vectorizer provided by scikit-learn, the TFIDF values for the propositions were generated and stored as a feature.

4.3.2 Lemmatisation

Each proposition was tokenized and stored in a Python list containing a word and its Part-of-speech (POS) tag obtained from the Natural Language Toolkit (NLTK)[1]. Each word and associated tag were then used as inputs for the NLTK wordnet lemmatizer. Providing the tag ensures that the correct root for each word was found, particularly in case with ambiguitity such as words that can serve as nouns or verbs. The lemmatizer was then passed as a parameter to the TFIDF vectorizer.

4.3.3 Part-of-speech tags

The spaCy Python library[2] was used with the en-core-web-sm model to measure the total number of POS tags within each proposition. The number of proper nouns, particles, verbs, nouns, adjectives, numerical symbols and symbols was recorded and used as a feature. Additionally, a boolean feature of 'contains a modal verb' records true if a modal verb is present within a proposition and false if not.

4.3.4 Quantitative Values

Quantitative values describing the propositions were measured and stored. For each proposition, the sentence length, average word length, individual word lengths, and longest word were measured and stored as separate features.

4.3.5 Sentiment

The sentiment intensity analyzer provided by the NLTK was used to measure the positive, neutral, and negative sentiment for each proposition. These were then stored as separate features.

[1] available at: https://www.nltk.org
[2] available at: https://spacy.io

4.3.6 Tense

The Stanford Parser[3] was used to count the occurence of past, present, and future tense verbs. A boolean feature for each tense was set to true if a verb of that tense was present.

4.3.7 Specificity

The Speciteller tool[4] was given a Python list containing the tokenized propositions as input. It then provided the specificity of each sentence which was recorded as a feature. The feature ranges from 0 to 1.

5 Results

	Precision	**Recall**	**f_1 score**	**Support**
Fact	0.63	0.62	0.63	58
Value	0.67	0.55	0.60	22
Policy	0.67	0.71	0.69	77

Table 3: SVM model results

The results achieved can be seen in table 3. A weighted average was measured using the f_1 score function available in scikit-learn showing an overall f_1 score of 0.65.

6 Conclusion and future work

This work has demonstrated an initial implementation of a ML model to automatically classify propositions. Using features including term frequency-inverse document frequency, Part-of-speech tags, and specificity, a SVM model was able to achieve an f_1 score of 0.65.
Although a promising start, it is clear that the work can be developed to achieve better accuracy results. The two main areas identified in which improvement could be gained are the use of additional data and additional or improvement of the features used.
Acquiring additional data, such as transcripts of other debates, would increase the number of examples that the Model can use. Currently, at 796 propositions, the data size is small and additionally, the split between proposition types is rather unbalanced. Resolving these issues would likely lead to an increase in model performance.
The other area for improvement, the features used, could by addressed by adding additional features, or making improvements to the currently used features. Additional features such as identifying speaker intent within a proposition, or looking for elements such as sarcasm or metaphors, may be worthwhile avenues for experimentation. Changes could also be made to the current features, for example, adjusting the tense identification feature to look at the sentence as a whole, instead of looking at the individual words within a proposition.

[3]available at: https://nlp.stanford.edu/software/lex-parser.shtml
[4]available at: https://www.cis.upenn.edu/ nlp/software/speciteller.html

References

[1] Jacky Visser, John Lawrence, Jean Wagemans, and Chris Reed. An annotated corpus of argument schemes in us election debates. In *Proceedings of the 9th Conference of the International Society for the Study of Argumentation (ISSA), 3-6 July 2018*, pages 1101–1111, 2019.

[2] Jean Wagemans. Constructing a periodic table of arguments. In *Proceedings of the 11th International Conference of the Ontario Society for the Study of Argumentation (OSSA)*, 2016.