Satire Detection

David Corney 15/10/2017

The task: Consider the problem of automatic satire detection. The input to the algorithm is a URL, and an output should be a score (0,1) denoting how likely the URL contains satirical content.

Part 1 (write-up): What would you use as an evaluation set? Can you split the satire detection problem into different subtasks, if so, which ones? Where would the signal come from? What limitations do you envision?

Part 2 (code): Create a baseline implementation for satirical content detection. Please use this example dataset: https://people.eng.unimelb.edu.au/tbaldwin/resources/satire/

Satire detection: what and why?

Satire comes in many forms, including exaggeration, parody, sarcasm and juxtaposition, all with the intention to humorously ridicule powerful figures or groups. If a satirical text is taken literally, it becomes a source of misinformation, and this can easily happen if removed from its original context. Satire can appear anywhere, and the scale of the web means that effective detection requires automation; the diversity and subtlety of sature requires sophisticated AI techniques.

Evaluation set

Having built a satire detector of any kind, we need to evaluate it on some new texts to predict how well it will perform. In the simplest case, this would be a set of documents, scraped from various websites, and manually labelled as being satirical or not. Tim Baldwin's "satire document collection"[1] is one such collection used in this study. In the accompanying paper, "Automatic Satire Detection: Are You Having a Laugh?"[2] by Burfoot & Baldwin (2009), the authors explain how they created the set. Briefly, they scraped articles from several known satirical sites (e.g. the Onion) and several serious news sites (e.g. AP). They then manually filtered out inappropriate articles, and edited the text to hide the source and its URL.

While this set is no doubt useful, it is quite distinct from the real-world challenge it represents in several ways. First, in practice, the URL of the site will always be known and is likely to be a very strong signal. Many serious news sites do publish at least some satire (e.g. The Borowitz Report); some deliberately mix satire with straight news (e.g. Private Eye). Nonetheless, knowing the URL would help to reliably identify many satirical pieces.

[1] *https://people.eng.unimelb.edu.au/tbaldwin/resources/satire/*
[2] http://dl.acm.org/citation.cfm?id=1667633

Second, the Baldwin set only compares ‘news’ and ‘satire’, ignoring other types of web content, such as ad copy from e-commerce sites; fiction; personal blogs; encyclopedias; academic papers and so on. By working with multiple classes in this way, a range of more specialist classifiers could be built, which together might be more effective than a binary classifier. Most web pages are neither satire nor news.

Third, only 233 satirical documents (and 4000 newswire documents) are in the set. This inevitably represents only a small part of the range of satire found online, especially as the texts were sampled from just 12 satirical sites.

Given more time and resources, a larger and more diverse set could be created by sampling more articles from more sites. Labels could be provided by crowd-sourcing, either though a purpose-built system or through existing tools such as Mechanical Turk and Crowdflower. In any case, each document should be labelled by multiple people as there is a degree of subjectivity in the decision: if 50% of readers think an article is satirical, does it make sense to claim that it is (or is not) satirical, even if the author claims it is? Such borderline cases will always exist, and any automated system should be aware of this, such as passing the final decision on to a human (‘human-in-the-loop AI’). A classifier built from a training set limited to those articles where multiple labellers agree on the label might allow full, confident automation of classification on a subset of documents, while referring the rest to people.

One way to make such a labelling task more efficient would be a form of active learning, where 1) an initial classifier is trained (e.g. using the Baldwin set); 2) it is presented with a large number of documents to predict labels for with confidence scores; 3) a subset of documents is chosen to be passed to human labellers, such that the labels will help improve the model by the greatest amount. For example, the documents where the classifier has the lowest confidence, or where the documents are closest to the classifier’s decision boundary, may be passed on for labelling.

Subtasks in satire detection

Potentially, different forms of satire could be detected independently. For example, there is a substantial body of research work in sarcasm detection, particularly for Twitter. Often, a series of specialist classifiers can be combined effectively to solve a broader problem.

One suggestion from Burfoot & Baldwin (2009) is that parodies often involve putting public figures in unlikely situations. They therefore suggest using named entity recognition (NER) to find combinations of entities that have very few matches on Google. Articles containing such combinations are more likely to be satirical, they suggest. Their own experiments show a small improvement to classification accuracy using this approach.

Below, we discuss differences between detecting satire at the level of sources, documents and sentences. Each of these could be used as a subtask with the results combined into a single prediction.

Signals for satire detection

Satire can potentially be detected at multiple levels, most obviously at the source level or the document level. For source-level detection, if a source is known to be (close to) either 0% or

100% satire, then all documents from that source can be labelled. This could be repeated for other unwanted content, such as comment pieces. Signals worth considering include:

- the URL, especially if partial white-lists or black-lists are available;
- keywords in the text or meta-data of the site. For example, the Onion's 'about' page states: "The Onion uses invented names in all of its stories, except in cases where public figures are being satirized";
- Limited number of articles and limited update rate. Mainstream news sites tend to produce rolling coverage with multiple updates of many stories each day. Specialist or low-budget news sites may be less heavily maintained, however.

In addition (or instead of) source-level satire detection, we can attempt document-level detection. This is important for sources with a more mixed (or unknown) output. Signals here include:

- Uncommon juxtapositions of named entities, as suggested in "Automatic Satire Detection: Are You Having a Laugh?". While they use a Google search for each set of entities, a regular large-scale pipeline of documents would allow a baseline frequency of co-occurrence to be stored and used to discover unusual combinations.
- Lack of similar stories elsewhere. Genuine news stories are more likely to appear in multiple sources in quick succession.
- Fact-checking the key claims. If the primary claim of a story does not appear in any existing knowledge base, then either it is breaking news not yet in the knowledge base, or it is some form of misinformation, possibly satire. However, it may be possible to learn from real news articles about the kind of events that do happen: famous people die; wars break out; company results are reported. Claims that fall outside of this scope may be flagged as unlikely to be true.
- The writing style of different authors has been successfully modelled by machine learning systems. It may be productive to learn to recognise several specific satirical authors and see if such as system can generalise to other authors with a similar style. Of course, many satirical writers deliberately imitate the style of serious news articles, so this may be difficult.

Within documents, it may be possible to train a sentence-level satire detector. An aggregation of scores for all sentences in a document could be used to classify the document itself; alternatively the most (or least) satirical sentences in a document could be shown to a human judge to make the final decision, without having to read the whole document.

Limitations:

Given the ambiguity of satire and its ever-changing face, no fully automated system will achieve perfect detection. It has often been claimed that 'satire is dead', as the news becomes more extreme and more unpredictable. If it is sometimes hard for a human to decide whether a story is satire or not, it will likely be impossible for a machine. (As an aside, it might be interesting to pass a range of historical news articles through a satire predictor to see if satire and mainstream news are indeed converging.)

Supervised machine learning approaches rely on labelled data sets, and these are expensive to create and maintain. As mentioned earlier, crowd-sourcing labels may be effective, and of course users of a tool that includes satire detection may provide feedback, explicit or implicit, which can be used to generate more labels.

Given enough documents to analyse, some will inevitably fall close to the decision boundary of a classifier, which cannot then make a confident prediction. These cases may then require manual filtering. This could still be machine-assisted, perhaps by automatically showing the most satirical sentences to the user (as mentioned above).

Implementation:

The code can be found at https://github.com/dcorney/satire**.** Some key functions are described here.

build_model() This uses the scikit-learn machine learning library to calculate the standard tf.idf values for terms in each document, and to train and evaluate an SVM classifier.

enhance_terms() This uses the spaCy NLP library to first identify named entities corresponding to people or organisations. It then uses spaCy's dependency parser to identify the corresponding noun chunks and the associated verbs. These are enhanced by simply appending multiple copies of these words at the end of the document. Because we are using a tf.idf representation, the word order does not matter, so the effect is simply to increase the significance of these terms. The motivation is that much satire is about famous or influential people or organisations, so the words corresponding to these targets and their actions are likely to be especially significant.

train_test() This is the top level function that calls others to load the data, enhance the documents, split into training and test (evaluation) sets, build and evaluate a classifier model, and display the results. For simplicity, we train and test on non-overlapping subsets of Baldwin's 'training set', and ignore his 'test set' of articles. This is unlikely to significantly change the results.

Results:

Here we summarise some example results using this code. In each case, we report the F1 score for the 'satire' class.

Baseline classifier (scikit-learn's 'dummy / stratified' approach): **F1 = 0.04**

SVM classifier (linear kernel, C=10) **F1 = 0.64**

SVM classifier (enhanced entities) **F1 = 0.67**

For all the SVM results observed, the precision was close to 1 with a recall of around 0.5. This is similar to the initial results in the Burfood & Baldwin (2009) paper. The baseline 'dummy' classifier here scores very badly, as it depends on the class frequencies and very few documents are labelled as satire.

While there is plenty of scope for improvement, we have reached the initial results reported by Burfood & Baldwin (though they improve these results by using better feature selection). We have also shown a small improvement by using a simple enhancement based on named entities and their context.

Conclusions:

Detecting satire is part of a wider goal of reducing misinformation and disinformation in a collection of news articles. A perfect satire detector could be used to reduce the risk of some types of misinformation, but is not enough in itself. To an end-user, different types of mistake may be more or less problematic. Put simply, is it a worse mistake to label a news article as satire, or vice versa? Presenting obviously humorously false information as if it were genuine is likely to undermine user's faith in the system, whereas perhaps falsely labelling a few genuine stories as suspect or fake is less critical, as other sources of the same stores are likely to appear and (hopefully) be labelled as genuine.