# Towards Computational Fact-Checking: Is the information checkable?

*Hugo Miguel Engrácio Farinha*

**Abstract.** Fact-checking has recently become a real world hot topic, especially in what concerns political claims. Several big players, such as, for example, Google or Facebook, have started addressing/making contributions to make "fact-checking" possible/available to the general public. However, most, if not all fact-checking platforms are largely manual, in the sense that most of the contributions and of the actual checking is performed by humans. Automatic computational fact-checking is still very far from being reliable and available on a large scale.

The current work is a contribute to the goal of automatic fact-checking by presenting features to distinguish checkable from uncheckable sentences and a fuzzy approach to computing sentence checkability, i.e., to answer the question: "is it possible to know if a sentence is worth to be checked?". Even though, this is a hot topic, few to none solutions have been presented to automatically assess the worthiness and liability of the verification of a sentence. The solution that is proposed is mainly based on Natural Language Processing methods, linguistics and fuzzy logic.

Assessing the checkability of a sentence can have many applications besides automatic fact-checking, like, for example: a broader fact-checking view (automatic checking of webpages and articles), the summarizing of information and the evaluation of factual information on texts. The main goal, however, was to focus on analysis of individual sentences, to provide an important tool to automatic fact-checking, by finding if a sentence is worth to, and can, be checked.

**Keywords:** Automatic Fact-checking, Sentence checkability, Fuzzy sets, Natural Language Processing, Linguistics.

## 1 Introduction: Is a sentence checkable? How checkable is it?

With the widespread of the internet, and the social web playing such an important role in peoples' lives, the necessity for automatic ways to validate the veracity of the information circulating in the web is increasing rapidly, and is currently a very hot topic. Automated Fact-checking is the name given to the process of, given a claim, a fact or, in general, a sentence, automatically assess its veracity based on publicly available sources of information in the internet. This work focuses on one of the first steps deemed necessary to achieve "Automated Fact-checking": measure the potential for a sentence "checkability".

As an example of a sentence that can be checked, or, its veracity can be assessed, is "Albert Einstein was born in 1879 in Germany". On the opposite side, "What was accomplished yesterday was nearly perfect" can't be checked, because, not only it's not easy to define what is "perfect" and what a "perfect event" is, but also it's impossible to define what a "nearly perfect event" is.

The exact meaning of "what is checkable" or "check-worthy" is not trivial. In this work we opted to consider that "checkable sentences" must be factoids [1] or exhibit similar characteristics. A factoid is a fact that has appeared in the news (in our case, a fact that can be checked in the internet). We propose an approach based on Natural Language Processing (NLP) and fuzzy sets to address a way to indicate how checkable a factoid is. The approach measures the potential of the quality of a future fact search and the accuracy of the veracity classification, i.e., the higher the checkability, the higher should be the chance to get a correct classification. As an example of the quantization of the checkability: the sentence "Albert was born in 1879" should score (and does score on the developed system) less than "Albert Einstein was born in 1879", since the search related to the latter will have more relevant information and the classification has more relevant information to check.

Since articles, news pages and webpages in general, usually consist of numerous sentences and claims, a broader view was also taken: quantize the sentences' "checkabilities" in such way that allows to order sentences according to their potential to be checked. With this, information in general, sentence, text or webpage, can be analyzed, the sentences with the most potential

can be extracted and the information can be checked in a more expedite way (opening the door to automatic verifying a whole webpage's veracity, extrapolating from the most checkable sentences). Interesting additional applications could also be:

- Summarize the information – given a text, extract the most checkable sentences, which in most of the cases coincide with the most important sentences;
- Evaluate the quality of a text (article or newspage, for example) – given the text, how many sentences are checkable? And how many have a high checkability? Articles written with basis on a high number of facts have more quality and should be more reliable than articles with less factual information.

The potential of this system is vast and the system that was developed paves the way for all these applications.

## 2 Related work

Even though fact-checking is currently a hot-topic, current technologies for full automated fact-checking are still either too dependent on "manual labour" or limited in scope.

Most of the research work in fact-checking addresses search and veracity classification. In this sub-topic it is possible to find several major players using different algorithms and (mostly manual) implementations: FightHoax [2] and ClaimBuster [3], in a beta state (based on Artificial Intelligence), WikiTribune [4] (the facts are verified by professional journalists), Google's Fact Check [5] (labels news or claims that are fact-checked by news publishers or fact-checking organizations) or politifacts [6] (claims classified by editors and reporters from the Tampa Bay Times).

Regarding the identification of check-worthy sentences, works are much sparser, and the only solution with some traction is ClaimBuster, which classifies a sentence's checkability in respect to "how factual it is" and "how relevant to the public it is if checked". However, ClaimBuster is very limited and its results highly questionable. For example, the claim "Donald Trump plans to build a wall dividing USA and Mexico" scores 22% in ClaimBuster [7]. The "check-worthiness" of a claim is relative and different for each person, but a score of 22% (almost not checkable) seems too low for a well-written and relevant present day claim.

## 3 Is the sentence checkable?

In this chapter, we present the methods for the first classification: assess if the sentence is "checkable" or "uncheckable". These methods are applied in a "Chain of Rules", which means that if any condition fails, the sentence is classified as not checkable. Sentences that fail this Chain of Rules are classified as "uncheckable" and are assigned a checkability of 0. The rules/methods are detailed on the following subsections.

### 3.1 Indefinite Pronouns/Adverbs and Incorrect Modality

The first rule that was used is conceptually rather simple: given a list of words, if the sentence is composed by any of those words, results in a classification of uncheckable. The lists of words are: indefinite pronouns (like "anything" or "something") [8], indefinite adverbs of time (like "always" or "rarely"), indefinite adverbs of degree (like "almost" or "nearly"), indefinite adverbs of manner (like "boldly" or "furiously"), indefinite adverbs of place (like "somewhere" or "elsewhere") [9], expressions that express incorrect modality for facts/factoids (like "the majority of cases", "probably" or "could") [10] and expressions used in expressing opinion (like "believe" or "his opinion").

This rule may seem too aggressive but it ensures that a checkable sentence is a factoid or a fact. Moreover, two features were taken into account to reduce the impact of incorrect results:

- The rules are applied only when the word exists on the sentence and has the respective Part of Speech ('NN'/noun, 'RB'/adverb, 'MD'/modal or all, in some cases);
- The list is totally customizable (allowing to remove and add words as needed) and, so, the filtering checkable/uncheckable depends only on the definition of "checkable".

### 3.2 Verb Conditions

This rule checks for the existence of a main verb on the sentence. Using SENNA [11] we obtain the Semantic Role Labeling of the sentence, that is, each verb and its arguments. So, for each subphrase of the sentence, if the Semantic Role Labeling did not find a verb, the sentence is classified as uncheckable.

Additionally, if after the lemmatization [12] of the verb, it does not exist in any of the databases (PropBank [13], VerbNet [14] or WordNet [15]), the sentence is also classified as uncheckable because the verb is invalid. This is related to the impossibility of performing a semantical analysis, which reveals that the sentence is incorrectly formed, and so, not verifiable.

There was the possibility for additional conditions on verbs, but it was decided not to implement them as they were too restrictive, even though they would make sense restricting checkable sentences to facts and factoids. These additional uncheckable verbs could be:

- Verbs in Gerund (like "running" or "writing") – used in actions that are happening in the present, making them, conceptually, hard to check, especially automatically;
- Verbs in Past Participle (like "written" or "driven") – used in actions that began on the past but are still occurring, for the same reason as the Gerund;
- Verbs in their base form (like "be" or "swim") – usually used together with some other verbs/expressions or in abstract sentences (like "I like to swim" or "I will be the first"), providing little value to the sentence's checkability if they are the main verb.

### 3.3 Arguments on Semantic Role Labeling

This is the rule conceptually more complex: based on the Semantic Role Labeling and on verb databases, the sentence is classified as checkable only if it has enough content (arguments) for the verb. The process is:

1. SENNA does Semantic Role Labeling on the sentence, identifying each verb and its arguments;
2. The verb is lemmatized;
3. We perform a search on PropBank, VerbNet and WordNet using the lemma of the verb to obtain the minimum number of arguments for the verb;
4. If the number of arguments on the sentence is larger than two (defined as the minimum for the sentence to have enough content to search) and larger than the minimum number found on the databases, the sentence is classified as checkable and uncheckable otherwise.

With this rule, we ensure that there is enough content on the sentence to link the action/verb to its arguments ("who did it?", "when?", "where?", "to who?", etc.) and, so, in a semantic point of view, ensure that it is possible to verify the veracity of the sentence.

## 4 Computing a sentence checkability

After assessing that the sentence is checkable, we propose to compute the sentence checkability based on the sentence's subjectivity, sentiment and opinion level, entropy and use of keywords. We make use of (mostly NLP) tools that are publicly available to compute the sentence scores regarding those factors, and use expert fuzzy knowledge to aggregate the information. Finally, we perform some syntax and semantic analysis to adjust the proposed checkability score. The following sections describe each step of the process.

### 4.1 Subjectivity and Sentiment/Opinion Score

By analyzing sentences used in claims extracted from articles and news, it was clear that sentiment and the expressing of opinions should be taken into account when computing how checkable a sentence is. For example, "Hotel X was very comfortable", reveals a personal opinion and, so, is considerably less checkable than, for say, "Hotel X has 120 rooms". Sentiment Vader [16], from the package NLTK, was used to assess the level of sentiment or opinion on a sentence. Since this method is dedicated to "Opinion Mining", which is not exactly the task we are addressing, we introduced a subjectivity classification in order to adapt and improve the results. NLTK Sentiment Utils [17] was used to perform the subjectivity classification. Additionally, a N-grams [18] approach was also used for a better normalization of the sentiment/opinion score (attenuate effects of single words on longer sentences).

The Sentiment/Opinion Score is used only when the sentence is classified as "Subjective" or this score is above a given threshold. Empirically, it was decided to use 0.6 as the Sentiment/Opinion threshold score.

### 4.2 Entropy and Keywords Score

Two other metrics deemed important to compute the checkability of a sentence are the amount of information within a sentence, and the use and importance of keywords. The more information on the sentence, the more information there is to check and the more definite the entities on the sentence are.

**Modified Shannon Entropy.** Computing the entropy is the most reasonable step to measure the amount of information on a sentence. Entropy uses probabilities of each symbol, or words in this case, to compute the desired measure. We used Shannon Entropy [19] with the equation (1).

$$H(sentence) = -\sum_{i=1}^{n} P(word_i)\log_b P(word_i)$$
$$P(x_i) = \frac{N_i}{\sum_{k=1}^{n} N_k}\ , with\ N_i = Number\ of\ occurrences\ of\ the\ word_i \qquad (1)$$

To improve results, not all of the words on the sentence were used. From the usual Bag of Words we excluded:

- Stop words from NLTK corpus (“the”, for example);
- Words (not numbers) with less than 3 characters (“Mr”, for example);
- All the punctuation (“,”, “;”,”!”, etc.).
- Punctuation was removed from the sentence before applying the method;
- Since RAKE does not give, for itself, score to numbers and numbers take a huge part in fact-checking (dates, values, percentages, amounts, etc.), the RAKE output was modified to 1.5 if the set is a number (i.e. numbers are worth slightly more than one word).

**Keywords Score - Modified RAKE.** Finally, we quantify the sentence’s keywords and their relevance. This score is a “dual” of the Entropy Score since it also measures the information on the sentence. However, with this score we try to measure the quality of the information instead of the amount of information.

To compute this score, a very simplistic approach was used: the RAKE method [20]. This method identifies keywords by going through the sentence and splitting it to remove stopwords. The method finds keywords clusters and computes a score for each one. The final Keywords Score is simply the sum of the clusters’ scores. Each score is computed based on: the number of words (consecutive non-stopwords) and the number of occurrences of each one on the sentence. To improve results the method was modified as follows:

**Normalization.** Shannon’s entropy and RAKE scores can take values between 0 and +∞, and there is no perception of what is an optimum or even a good value in what concerns measuring a sentence checkability. With this in mind, we devised a normalization procedure for both scores using several training sentences containing different amounts of information (manually classified with a desired score). *Fig. 1* shows the normalization result for the training {computed value, desired score} pairs. The computed entropy was represented in $2^{H(x)}$, with H(x) being the entropy computed using (1). Linear interpolation is applied to obtain a score to any computed value. As a result of the normalization procedure we obtain fuzzifiable scores.

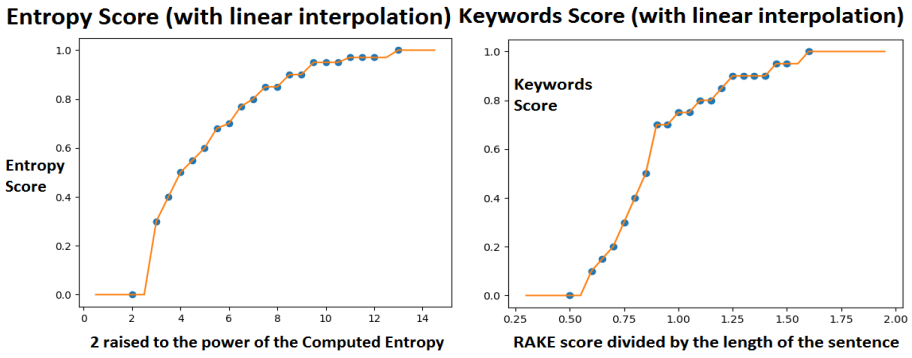


**Fig. 1.** Normalization of the computed Entropy and RAKE scores

### 4.3 Fuzzy Score Aggregation

The aggregation of the computed scores is done using expert knowledge expressed by a Fuzzy If-Then Mandani rule base (implemented using [21]). Fuzzy Sets were chosen because of the following reasons:

- Scores are normalized and can take any value between 0 and 1;
- The several thresholds, parameters and functions used to manipulate, adapt and compute the individual scores, were used based on testing and expert knowledge, and are not proven to be optimal. Therefore, the method to be used should be robust and not change drastically with further changes of these defined parameters;
- Expert knowledge was easy to implement and gave sound results.

We defined three membership functions for the inputs (Low, Medium and High). Experts found it difficult to use more than three linguistic terms when devising the rules for this problem. We used five membership functions evenly distributed through the Universe of Discourse (VL, L, M, H, VH) to indicate Checkability. Trapezoidal functions provided the best trade-off between complexity and accuracy of the result (Triangular was found to be slightly worse). The membership functions were defined as represented in *Fig. 2*.

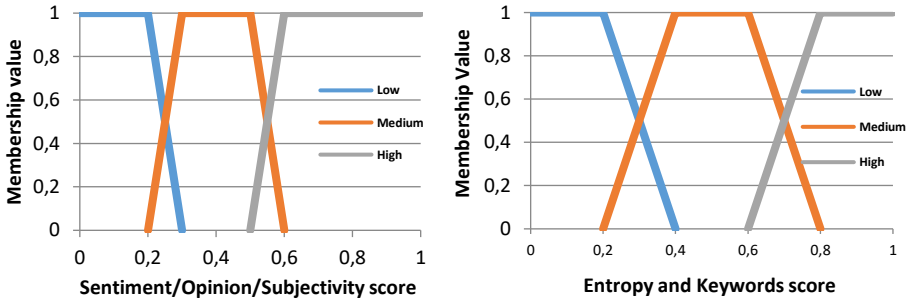


**Fig. 2.** Membership functions for the three input scores

*Table* **1.** shows the fuzzy rulebase used to aggregate the computed scores. Each line contains one rule of the type "IF Sentiment/Opinion score is *A*, and Entropy score is *B*, and Keywords score is *C*, THEN Checkability is *Z*", where *A*, *B*, *C* and *Z* are fuzzy linguistic membership functions. The rules were defined based on expert knowledge and annotated data examples.

**Table 1.** Fuzzy Rulebase used for score aggregation

| Sentiment/Opinion | Entropy | Keywords | Checkability |
|---|---|---|---|
| High | - | - | Very Low |
| Medium | Low | - | Very Low |
| Medium | - | Low | Very Low |
| - | Low | Low | Very Low |
| Medium or High | Medium or High | Medium or High | Low |
| Low | Medium | Low | Low |
| Low | Low | Medium | Low |
| Low | Medium | Medium | Medium |
| Low | Low | High | Medium |
| Low | High | Low | Medium |
| Low | Medium | High | High |
| Low | High | Medium | High |
| Low | High | High | Very High |

### 4.4 Syntax and Semantic adjustment

Some additional metrics based on the syntax and the semantics of the sentence were added in order to further improve the obtained score.

We identify valid/invalid temporal expressions or locations using Semantic Role Labeling [22] based on SENNA [11]. The reasoning behind this procedure is that sentences that refer to a non-existing location or time, are obviously less checkable. After SENNA identifies such expressions, we try to validate them with Ternip (temporal expressions) [23] or a custom database built based on GeoNames [24] and Google Places (locations) [25]. The sentence checkability is penalized by 40% if all expressions are invalid, or increased by 10% if an expression is valid.

Then we check whether after applying Coreference Resolution, performed using CoreNLP [26], the sentence still contains personal pronouns. If affirmative, i.e., CoreNLP could not find the entity to link the pronoun, then the sentence's checkability is decreased by 40%.

Finally, we identify indefinite entities, such as, for example "The ship …". CoreNLP does a syntactic analysis [27] on the sentence that is used to group Noun Phrases (NP). In each group, each common noun is verified for the existence of articles (such as "a") or adjectives (such as "several") before the noun and some words after the noun that can describe it. If there is an article or adjective before the noun but no following words (on the same NP group), the sentence's checkability is decreased by 40%.

## 5 Examples and comparison with ClaimBuster

The described approach has been implemented and tested in a large range of sentences. Results are sound, very encouraging and compare very favorably against ClaimBuster [7]. *Table 2* shows some examples of the computed checkability.

**Table 2.** Some examples and comparison with ClaimBuster

| Sentence | ClaimBuster Score | Computed Checkability |
|---|---|---|
| "HSBC lays off 120 technology staff in Hong Kong in cost-cutting plan" | 37% | 100% |
| "Thomas Edison invented the first commercial light bulb in 1879" | 35% | 100% |
| "Donald Trump plans to build a wall dividing USA and Mexico" | 22% | 88% |
| "In state after state polls make clear that the American public understands the Kelo ruling is a disaster" | 36% | 61% |
| "Lisbon has a weak system of public transports" | 18% | 27% |
| "Seven US Navy crew members are missing after their ship collided with a merchant vessel off the coast of Japan." | 45% | 55% |
| Previous sentence without the period (the score should be equal) | 39% | 55% |
| "Harry Davis 30 or 40 potatoes in percent 300 350 and 400 percent." | 75% | 0% |
| "The first Sherlock Holmes was written in 1887 by Sir Arthur Conan Doyle" | 47% | 96% |
| "Several tourists were impressed by the ship." | 31% | 6% |

Looking at the examples, it is observable that even for facts, completely checkable and correct, ClaimBuster has an output that is extremely low. For example, the sentence "Thomas Edison invented the first commercial light bulb in 1879" has an output of only 35% on ClaimBuster, significantly inferior to the checkability assessed by the system (100%). To further exemplify the flaws of this solution, it is observable that this fact is classified with the same checkability as the sentence "Several tourists were impressed by the ship.", which makes absolutely no sense given the low factual information on the sentence.

The only sentence that achieved a high score was "Harry Davis 30 or 40 potatoes in percent 300 350 and 400 percent." which is meaningless and has no content to check, no verb but it has a named entity and several numbers, which gives the impression that ClaimBuster's scores are somehow proportional to the amount of words like "percent", numbers and of named entities on a sentence.

It is impossible to conclude which solution is correct, but the developed system seems more sound and able to translate better the checkability to its real world meaning: the potential to check the veracity of a sentence.

## 6 Validation with dataset and annotations

The developed system was also tested for validation with a dataset of 200 sentences with different characteristics and testing a wide diversity of situations. The dataset targeted mainly three types of content: news headlines, common sentences from dialogues and facts. The system was compared with annotations from three students (from Biomedical Engineering and Computer Science), from people on CrowdFlower [28] (platform where rows of data are given for classification to random people) and from the developer. To facilitate the annotations, four categories were defined: "uncheckable"/"Not Checkable", "low"/"Checkable but with difficulties", "medium"/"Checkable" and "high"/"Easily Checkable" and the system's thresholds were defined as 0.3 (low to medium) and 0.7 (medium to high). The metrics that were computed from the results are presented on *Table* 3.

**Table 3.** Metrics computed from the results of the validation with the dataset

| | System vs CrowdFlower | System vs Person 1 | System vs Person 2 | System vs Person 3 | System vs developer |
|---|---|---|---|---|---|
| Concordance | 33.50% | 46.50% | 47.00% | 54.00% | 74.50% |
| Almost concordance | 85.50% | 81.00% | 86.50% | 90.00% | 98.00% |
| Average score difference | 25.61% | 24.59% | 22.27% | 19.55% | 12.54% |
| False positives | 9.74% | 16.88% | 12.34% | 11.04% | 2.60% |
| False negatives | 63.04% | 39.13% | 45.65% | 36.96% | 17.39% |

Regarding the metrics that were obtained, it is clearly observable that, comparing with CrowdFlower, the results were substantially better comparing the system to the developer's classifications, with the three persons having intermediary results. This happens because of the deeper understanding on the subject, on what is intended and because each classification was carefully made, whereas the classifications from CrowdFlower were made with only a few guidelines (and assumptions) and as part of a routine.

Despite these differences, the system still reveals concordance with the human annotators, with a complete concordance in category in 33.50% to 54% of the sentences and almost concordance (same category or one category of difference) in 81% to 90% of the sentences. Given the complexity of the concept and difficulty of deciding between categories, these results are sound and reveal that the system tends to follow human annotations.

Regarding false positives and false negatives, it's notable that the system discards (classifies as uncheckable) much more sentences than human annotators. This happens because of the restrictive definition of a "checkable sentence", which discards sentences that leave doubt about their checkability, with people still classifying them as checkable. However, the false negatives still happen only in 36.96% to 63.04% of the cases. Focusing now on the false positives, the value is much lower, only happening in 9.74% to 16.88% of the cases, revealing that the system tends to follow the people's opinions on checkable sentences.

## Future Work and Conclusions

Despite the interesting results, the proposed approach still has room for improvement, especially in what concerns the optimization of the fuzzy sets and rulebase. Other possible improvements include the fuzzification of the Syntactic and Semantic adjustments and review of all the methods used on the system.

To conclude, the system that was developed identifies several important features when dealing with the concepts of uncheckable/checkable and checkability. It provides an important tool on the hot topic of Automatic Fact-Checking and for additional applications like summarization of information and evaluation of factual information on a text. The results that were obtained prove that the system is reliable and its classifications tend to be in concordance with people's classifications.

Furthermore, even being only a proof of concept, the results that are obtained seem to be much more sound than the results of ClaimBuster, which is the only alternative solution to this system.

## References


[1] D. Hiskey, "Difference between fact and factoid," 2010. [Online]. Available: http://www.todayifoundout.com/index.php/2010/02/the-difference-between-a-fact-and-a-factoid/. [Accessed: 20-Sep-2017].

[2] V. Tzekas, "FightHoax," 2017. [Online]. Available: http://fighthoax.com. [Accessed: 12-Jun-2017].

[3] N. Hassan and M. Tremayne, "Toward Automated Fact-Checking : Detecting Check-worthy Factual Claims by ClaimBuster," 2017.

[4] "Wikitribune," 2017. [Online]. Available: https://www.wikitribune.com. [Accessed: 10-Jun-2017].

[5] J. Kosslyn and C. Yu, "Google Fact Check," *Journalism & News, Google*, 2017. [Online]. Available: https://blog.google/products/search/fact-check-now-available-google-search-and-news-around-world/. [Accessed: 12-Jun-2017].

[6] "Politifact," 2017. [Online]. Available: http://www.politifact.com/truth-o-meter/. [Accessed: 11-Jun-2017].

[7] "ClaimBuster Demo," 2017. [Online]. Available: http://idir-server2.uta.edu/claimbuster/demo. [Accessed: 22-Sep-2017].

[8] W. Smith, "Indefinite pronouns," *K12Reader*, 2016. [Online]. Available: http://www.k12reader.com/term/indefinite-pronouns/. [Accessed: 06-Jun-2017].

[9] J. Essberger, "English Grammar," *EnglishClub*, 2005. [Online]. Available: https://www.englishclub.com/grammar/. [Accessed: 23-May-2017].

[10] University of Sydney, "Modality," *The Write Site*, 2012. [Online]. Available: http://writesite.elearn.usyd.edu.au/m2/m2u4/m2u4s4/m2u4s4_3_5.htm. [Accessed: 20-May-2017].

[11] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural Language Processing (Almost) from Scratch," *J. Mach. Learn. Res.*, vol. 12, pp. 2493–2537, 2011.

[12] D. Jurafsky and J. Martin, "Lemmatization," in *Speech and Language Processing*, 3rd ed., 2017, p. 2.

[13] E. Loper, "PropBank Corpus Reader," *NLTK Documentation*, 2017. [Online]. Available: http://www.nltk.org/_modules/nltk/corpus/reader/propbank.html. [Accessed: 20-May-2017].

[14] E. Loper, "VerbNet Corpus Reader," *NLTK Documentation*, 2017. [Online]. Available: http://www.nltk.org/_modules/nltk/corpus/reader/verbnet.html%0A. [Accessed: 20-May-2017].

[15] S. Bethard *et al.*, "WordNet Corpus Reader," *NLTK Documentation*, 2017. [Online]. Available: http://www.nltk.org/_modules/nltk/corpus/reader/wordnet.html. [Accessed: 20-May-2017].

[16] C. J. J. Hutto and E. Gilbert, "Vader: A parsimonious rule-based model for sentiment analysis of social media text," *Eighth Int. AAAI Conf. Weblogs …*, pp. 216–225, 2014.

[17] B. Pang and L. Lee, "A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts," 2004.

[18] D. Jurafsky and J. Martin, “Language Modeling with N-grams,” in *Speech and Language Processing*, 3rd ed., 2017.

[19] D. Ueltschi, “Shannon Entropy,” in *Teaching*, 2010.

[20] S. Rose, D. Engel, N. Cramer, and W. Cowley, “Automatic Keyword Extraction from Individual Documents,” in *Text Mining. Applications and Theory*, M. W. Berry and J. Kogan, Eds. John Wiley and Sons, Ltd, 2010, pp. 1–20.

[21] J. Warner *et al.*, “Scikit-Fuzzy,” Oct-2017. [Online]. Available: https://doi.org/10.5281/zenodo.1002946. [Accessed: 05-Jun-2017].

[22] D. Jurafsky and J. Martin, “Semantic Role Labeling,” in *Speech and Language Processing*, 3rd ed., 2015.

[23] C. Northwood, “Ternip – Temporal Expression Recognition and Normalization,” 2016. [Online]. Available: https://github.com/cnorthwood/ternip. [Accessed: 04-Jun-2017].

[24] “GeoNames,” 2017. [Online]. Available: http://www.geonames.org/export/. [Accessed: 25-May-2017].

[25] “Google Places,” 2017. [Online]. Available: https://developers.google.com/places/web-service. [Accessed: 10-Jun-2017].

[26] K. Clark and C. D. Manning, “Entity-centric coreference resolution with model stacking,” *Assoc. Comput. Linguist.*, pp. 1405–1415, 2015.

[27] D. Jurafsky and J. Martin, “Syntactic Parsing,” in *Speech and Language Processing*, 3rd ed., 2017.

[28] “CrowdFlower,” 2017. [Online]. Available: https://www.crowdflower.com. [Accessed: 15-Jul-2017].