Jumping Finite Automata for Tweet Comprehension

Stephen Obare  
School of Computing & IT  
Jomo Kenyatta University of Agriculture and Technology  
Nairobi, Kenya.  
smobareo@gmail.com

Abejide Ade-Ibijola  
School of Consumer Intelligence and Information Systems  
University of Johannesburg  
South Africa  
abejideai@uj.ac.za

George Okeyo  
School of Com Sci & Informatics  
De Montfort University  
The Gateway, LE1 9BH  
Leicester, United Kingdom.  
george.okeyo@dmu.ac.uk

Abstract—Every day, over one billion social media text messages are generated worldwide, which provides abundant information that can lead to improvements in lives of people through evidence-based decision making. Twitter is rich in such information but is faced with a number of challenges which include ambiguity of the language in which tweets are written, high volumes, large number of spelling and grammatical errors, abbreviations, slang, meaningless information, the use of improper sentence structure [5], [7]. Manually sifting through these massive volumes of data to find information that is most useful is time consuming. Machine learning techniques have been used to comprehend tweets with varied results [4]. Supervised learning techniques have primarily been applied in detecting small scale events e.g., civil unrests with the requirements of costly non-automatic data labeling which is labour intensive and time consuming [19]. Comprehension tasks that have no readily available labeled training data, are voluminous and noisy in nature like Twitter is the norm in todays computing environment and not the exception [6].

For detecting general and large scale events, unsupervised learning techniques which involves a set of documents with no further knowledge about the set has been used [27]. A model built upon such a set tries to find similarities and differences between the documents and separates them into clusters, where documents within each cluster are as similar as possible, and as different as possible between the clusters [26]. These clusters, however, do not have any real meaning. They are build on the observed features of the document set. One needs to interpret them to get useful results [24].

Beyond the scalability issues, the efficiency requirement is a pragmatic challenge for tweet processing. The processing speeds of existing NLP tools are often not up to the data generation speed e.g., 650 million Twitter messages per day4. As a result, the efficiency gap between data generation and processing restricts the effectiveness of Twitter data processing for many real world applications. Automatically extracting actionable information from this type of data is an active research area in the domain of natural language processing (NLP) [25].

This paper presents a new approach based on formal language and automata theory to the tweet comprehension problem. The approach includes a Kenyan WordNet for

improving the utility of tweets through ambiguity resolution, a formalized space of tweets variations and a JFA based tool that takes raw tweets, recognizes events of interest (syntax and semantics) and uses the derived insights to annotate maps. As a result this paper makes three key contributions:

1) WoLK - a novel lexical semantic knowledge base for Kenyan language that bridges the gap between WordNet\(^5\) and domain knowledge,
2) Formalized tweet space - generated a search space of possible variants of valid tweets that correctly identifies an event of interest, and
3) JFA - adapted methods from formal aspects of computing that are optimal for performing tweet comprehension task.

The rest of this paper is organised as follows. We present related work in Section 2. Section 3 specifies the design of WoLK and formalization of the space of tweet variation for abstraction onto a Finite Automata (FA). Section 4 demonstrates Automata-Aided ATC, a new tool that implements our technique while Section 5 presents conclusion and future work.

2. Related Work

This section presents a review of a number of language resources used in extracting semantics and techniques used in estimating similarities between sentences which are important for this research work.

2.1. WordNets

Over the past decades, the research community in the area of NLP have proposed a number of approaches for assigning senses to words based on the context of a sentence [32]. The approaches have lately been grouped into two main methodological approaches: knowledge-based and corpus-based algorithms [33]. Knowledge-based algorithms use lexical semantic resources to disambiguate words by defining explicit sense distinctions for assigning the correct sense of a word in context. Knowledge based algorithms give higher precision in disambiguating words in context but suffer from overlap sparsity and their performance depends largely on accuracy of dictionary definitions. Corpus-based methods use machine-learning algorithms which can either be supervised or unsupervised to disambiguate words from available sense inventory and annotated corpora for the case of supervised learning and in the case of unsupervised learning where sense inventory and annotated corpora is not required. Both knowledge-based and corpus-based algorithms present different benefits and drawbacks.

For knowledge based approaches, WordNet, which is a lexical semantic resource providing information about words with their meanings, has been widely used [4]. WordNet is currently the most advanced a lexical database created manually by English linguists providing an effective combination of traditional lexicographic information and modern computing.

In terms of structure, the main relation among words in WordNet is synonym. Nouns, verbs, adjectives, and adverbs are grouped into sets of cognitive synonyms called synsets which describes a distinct concept. Several other relations exist between synsets and words, such as antonymy hyponymy and meronymy as shown in Figure 1.

![Figure 1: Structure of WordNet, Adapted from [9]](image)

As observed by [10], manual construction of WordNet is a time consuming task and requires linguistic knowledge. In order to achieve comprehensive WordNet in languages other than English, two main approaches have been used [11]:

a). Merge approach – in this technique, an exhaustive repository of senses (meanings) of each word is compiled, synsets are then created that contain all of the applicable words for a given sense [12].

b). Expansion approach – existing synsets from a reference WN are used as a guide to create corresponding synsets in a new WN, by gathering applicable words that represent the meaning of the synset. This approach has been shown to be suitable for under resourced languages [13].

Two methods have generally been used in expanding WordNets: automatically [17], and semi-automatically [18]. We highlight a number of WordNets developed based on expansion approach that are relevant to the development of a WordNet for the language of Kenya used in this work. The WordNets include: EuroWordNet developed by linking several European languages to English WordNet [14]; Persian WordNet [15]; Finnish WordNet [15]; Polish WordNet [16] and African WordNet (AWN) [13] created by aligning several languages spoken in Southern Africa. A number of tools were used in development of AWN including DEBVisDic\(^6\) editor tools for linguists building AWN.

2.2. Estimating Sentence Similarity

In how many ways can the same event be reported on Twitter? This is the question of variability [34]. Creativity is

---

5. https://wordnet.princeton.edu/

highly involved in tweeting and two users are often likely to submit two different tweets [37]. If variability can be studied and modeled, then it becomes possible to build systems for comprehending tweets automatically, because such a system will be aware of every possible variation, and hence, will have the knowledge of the space of solutions.

[20] proposed a method to find similarities in sentences using Semantic Nets and Corpus Statistics. Their method was evaluated to be best suited with short lengths. Sentence semantic similarity calculating method based on segmented semantic comparison was proposed in [30]. The achieved best results in short sentences. [28] described a metric method for computing sentence level semantic textual similarity based on a probabilistic finite state machine model that computes weighted edit distance. [8], [22] estimated context similarity based on closeness of semantic load of two comparing sentences. They built an FA through a systematic analysis of the patterns of meaning and use for each verb.

In this work we construct WoLk, a WordNet for the language of Kenya which consists of three parts: Princeton WordNet, local lexical dictionaries, (Kiswahili and Sheng) and annotated corpus in order to develop an initial lexicon for the language of Kenya which consists of three parts: Princeton WordNet, local lexical dictionaries, (Kiswahili and Sheng) and annotated corpus in order to develop an initial lexicon.

3. Semantics and Comprehension

In this section, we describe the major components that constitute our technique.

3.1. Definitions

Definition 1 (Symbol, Alphabet, and String [21]). A symbol is a single token or word. An alphabet, denoted by \( \Sigma \), is any finite set of symbols (words). A string is formulated from concatenation of zero or more symbols (words).

Definition 2 (Lexemes and Lexical Analysis [25]). Lexical analysis is the process of reading tweets and grouping them into “lexically meaningful” tokens referred to as lexemes.

3.2. WoLK: A WordNet for the Language of Kenya

We developed a framework for resolving ambiguity in tweets by first examining our Twitter corpora, paying particular attention to unknown tokens. Our experiment involved 28,361 which we manually annotated revealed that 12,613 tokens (44.47%) were regarded as unknown by WordNet, out-of-vocabulary (OVV) and 14,168 tokens (49.96%) were considered as in-vocabulary (IV) indicating the need to integrate more domain knowledge in WordNet knowledge base.

3.3. Modelling Variability in Tweets

A simple event of interest can be reported in quite a number of ways, each unique way being a mere variation resulting from valid permutation of tokens of interest. We formalised a novel approach to determine the variations (alternate and equivalent) of any given tweet that helps us to automatically determine if a new tweet describes the same event.

Let \( \{S_1, S_2 ... S_n\} \) be a set of tweets reporting the same event.

\[
S_1 \equiv S_2 \equiv ... \equiv S_n
\]

if and only if they are recognized by some automata.

\( S_1 \) is the base tweet (the primarily known event tweet from which other equivalent tweets are compared and/or derived). If \( twt_{(s1)} \) is the base tweet and \( \Sigma_s \) the alphabet of semantic tokens, then the language of all solutions to the tweet comprehension problem (which includes the base tweet), over \( \Sigma_s \) is given as:

\[
L_s = \{twt_{(s1)}\} \cup \{L(twt)\}
\]

\[
L_s = \{twt_{(s1)}\} \cup \{twt_1, twt_2, twt_3, ..., twt_n\}
\]

We present an algorithm based on the formalisms that creates a space of alternative solutions to the entire tweet comprehension problem by computing the concatenation of tokens in tweets. The space of solutions generated by the new algorithm represents the alternative ways a user may uniquely write a tweet describing the same event but making different choices of language constructs while exercising his/her limited or extensive knowledge of the rules of discourse. The idea is to take the seed and then generate variations based on the seed from the product of its parts (semantic tokens).

Algorithm 1 Generating Tweet Variations (n)

Require: A tweet string of length \( n \) where \( n \geq 0 \).

Ensure: Number of variations for tweet string of length \( n \).

1. for \( i = 1 \) to \( n! \) step \( 1 \) do
2. begin
3. for \( j = 1 \) to \( n \) step \( 1 \) do
4. begin
5. \( \text{int } m = (n - j)! \)
6. \( \text{divide}(i, m) \)
7. if \( i \neq 0 \)
8. then
9. put \( j \) to the \((m + 1)_{th} \) empty position
10. else
11. put \( j \) to the \(m_{th} \) empty position
12. end for
13. output one permutation
14. end for
3.4. Abstracting the Space of tweet Variation

In [39], abstraction is described as the process of removing characteristics from a data set, in order to reduce it to a set of essential characteristics. We abstract the space of tweet variations onto an FA as a step towards comprehending tweets.

3.4.1. The States. We represented states by alphabets. Based on works by [31], events involve various participants and attributes and form a semantic (argument) structure: who did what to whom where and when. We represent a sample of these structure that forms our semantic tokens below. We identified these five entities that make up our alphabets in each tweet from the corpus.

Given that:

\[\sum \text{who} = i, \sum \text{what} = j, \sum \text{whom} = k, \sum \text{where} = l, \sum \text{when} = m\]

Our state can therefore be treated as the combination of symbols/alphabets, whether single or multiple which is \(a_i, b_j, c_k, d_l, e_m\).

Three gangsters were shot dead by policemen at Ongata Rongai
Two gangsters were shot dead near Langata by police men?
City MCA had dinner before killing his wife in a horror incident
Car jacking incident in Nyeri
Police kill a gangleader in Nairobi
A Woman has been sentenced to death for trafficking drugs in Gesonso Kisii county.
Youths are barricading the road at Southlands affecting motorists

3.4.2. Transition Process. Each tweet from Section 3.4.1 contains \(\Sigma\). As depicted in Figure 2, we modelled our transition process using graph traversal.

3.5. Comprehension using JFA

We integrated the concepts described in Sections 3.2, 3.3 and 3.4 in our JFA technique. Within a particular running process in JFA, a computational step may be performed anywhere within a tweet string [21]. Therefore, before the next step is carried out, the process may jump over a large portion of the tweet string to the desired position of execution.

<table>
<thead>
<tr>
<th>JFA symbols</th>
<th>a_i (who)</th>
<th>b_j (what)</th>
<th>c_k (whom)</th>
<th>d_l (where)</th>
<th>e_m (when)</th>
</tr>
</thead>
<tbody>
<tr>
<td>a_0 Three gangsters</td>
<td>b_0 shot dead</td>
<td>c_0 police men</td>
<td>Ongata Rongai</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a_1 Magondu</td>
<td>b_1 battered</td>
<td>c_1 karao</td>
<td>Umoja</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a_2 Youths</td>
<td>b_2 barricade road</td>
<td>c_2</td>
<td>Kisii county</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a_3 MCA</td>
<td>b_3 kills</td>
<td>c_3 wife</td>
<td>in town</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a_4 Lady’s bag</td>
<td>b_4 snatched</td>
<td>c_4 thugs</td>
<td>Karen</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a_5 Thugs</td>
<td>b_5 terrorize residents</td>
<td>c_5</td>
<td>Jamhuri High School</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a_6 Student</td>
<td>b_7 stabbed</td>
<td>c_6 fellow students</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>a_7 Ex-GSU</td>
<td>b_8 harassment</td>
<td>c_7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>a_8 Cop</td>
<td>b_9 steals</td>
<td>c_8 motorbike</td>
<td>Kiambu</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

TABLE 1: JFA symbols table

The tweet of Example 1 contains five JFA symbols (semantic tokens) from the symbols table that are important
for our comprehension task namely:

a. magondis, b. gunned, c. karao, d. Umoja.

These tokens can be matched to the set of alphabets \( \{a, b, c, d\} \)

**Example 1.**

Three suspected magondis gunned in Umoja Nairobi in shootout with Karao arrived...
AK-47 magazine, bundee, ignition switches recovered

Following works by a number of researchers on the effect the following factors when reporting an event: [35] on effect of spelling, [36] on effect of input devices, and [38] on effect of community, the tweet of Example 1 can be re-written as shown in Example 2:

**Example 2.**

This is getting out of control. kupigwa ngeta na wagondi in between the dark alleys in Umoja.
Three shot today by police, after stealing car.
The needs to be cleared of thugs ASAP!....

We use (WoLK) developed in Section 3.2 as a comprehensive language resources for Kenyan language providing meaning to a number of tokens such as wagondi that helps us decipher that tweet 1 and 2 refer to the same event.

Given the sample tweet in Example 1 and based on our new formalization of variability described in Section 3.3, the space tweet variability can be given as

\[ \Sigma \text{twt}_{i}, i > 0 = 4! = 24 \text{ possibilities.} \]

Abstracted on a JFA

\[ M = (\{S_0; S_1; S_2; S_3; S_4\}, \{a_i, b_j, c_k\}, \{0, r, s\}; \{S_4\}) \]

States - \( \{S_0; S_1; S_2; S_3; S_4\} \),
Alphabets - \( \{a_i, b_j, c_k\} \),
Finite set of rules - \( R \),
Start state - \( s \), and
Accept state(s) - \( \{S_4\} \).

With

\[ R = \{S_0a_i \rightarrow S_1, S_1b_j \rightarrow S_2, S_2c_k \rightarrow S_3, S_3c_k \rightarrow S_4\} \]

with the transition

\[ b_ja_icjb_jc_kS_0a_i \rightarrow b_ja_icjb_jc_kS_1b_jc_k [S_0a_i \rightarrow S_1] \]
\[ \rightarrow b_ja_icjb_jc_kS_1b_jc_k [S_1b_j \rightarrow S_2] \]
\[ \rightarrow b_ja_icjb_jc_kS_2c_k [S_2c_k \rightarrow S_3] \]
\[ \rightarrow b_ja_icjb_jc_kS_3c_k [S_3c_k \rightarrow S_4] \]

\( L(M) \) recognises/accepts the tweet string of Example 1.

\( L(M) = \{w \in \{a_i, b_j, c_k\}^* : |a_i| = 1 \leq |b_j| \leq 3 = 1 \leq |c_k| \leq 3\} \)

\( a_i; b_j; c_k \) are defined in the JFA table of tokens extracted from our tweet corpus.

**Figure 3:** Transition Diagram tweet string in Example 1

**Figure 4:** ATC System Architecture

### 4. Implementation

#### 4.1. ATC Tool

We describe components of our approach which we developed for automating tweet comprehension. The essence of the system, as depicted in Figure 4, is an underlying preprocessing module which filters out noise before comprehension by a repository of JFAs.

Preprocessed tweets are then passed through a repository of JFAs for comprehension. Depending on the event of interest, the list of relevant tweets are then stored in a database of tweets of interest. The process is briefly illustrated in Algorithm 3.

#### 4.2. Testing

We tested ATC with tweets retrieved from Nairobi city between September and December 2017. This history covers a period of four months and comprised 31,225 tweets from various crime hashtags. After preprocessing, (filtering retweets, stop word, abbreviations elimination, resolving slang using WoLK), the number of tweets reduced to 11,651.
Algorithm 2 Text Normalisation

1: function NORMALISE_INPUT(user_input, acceptance_rate)
2:     for each user_text in user_input do
3:         if user_text.Length is greater than 5 then
4:             set dictionary ←− load_WoLK()
5:             if WoLK.contains(user_text) is false then
6:                 for each dict_word in WoLK do
7:                     if calculateLevenshtein(word, user_text) >= acceptance_rate then
8:                         return word
9:                     ExitFor
10:             else
11:                 return user_text
12:         end if
13:     end for
14: end function

Algorithm 3 ATC Algorithm

1: function JFA_PARSER(raw_tweet, jfa_repository[ ], threshold) returns status
2:     status ←− {failed, 0.0} // {parsing status, percentage matched}
3:     set raw_tweet ←− preprocess_input (raw_tweet)
4:     set matched_jfas to 0
5:     convert raw_tweet to raw_tweet_array
6:     for each jfa in jfa_repository do
7:         for each raw_tweet_token in raw_tweet_array do
8:             if (jfa contains raw_tweet_token) OR (jfa contains synonym (raw_tweet_token)) then
9:                 POP raw_tweet_token from jfa
10:             end if
11:         end for
12:         if jfa is empty then
13:             increment matched_jfas by 1
14:         return status
15:     end for
16: return status

4.3. Results

We present the performance analysis of ATC, based on the 11,651 tweets from the perspective of accuracy in recognising tweets of interest. ATC failed to recognize only 1,631 hence the recognition accuracy of ATC is 86%. Not all tweets that were not recognized are crime tweets and we attributed the failure to recognize some crime tweets to the non-exhaustive nature of WoLK knowledge base.

5. Conclusions and Future Work

The ATC tool provides automated tweet comprehension platform for tasks that require shifting through massive volumes of unstructured data. We started by developing a novel lexical semantic knowledge base to help us increases the coverage of WordNet and interpret tokens of interest, formalized the space of tweet variation and abstracted this space on a JFA as core components of ATC tool.

Tested on 31,225 tweets, ATC recognised 86% of our test data set. The 14% unrecognized tweets is significant and we will expand the coverage of ATC so that it can understand more of the event tweets.

References


[20] Li, Yuhua and McLean, David and Bandar, Zuhair A and O’shea, James D and Crockett, Keeley: Sentence similarity based on semantic nets and corpus statistics, journal=IEEE transactions on knowledge and data engineering, volume=18, number=8, pages=1138–1150, year=2006, publisher=IEEE


[29] Carpenter, Patricia A and Miyake, Akira and Just, Marcel Adam: Language comprehension: Sentence and discourse processing. Annual review of psychology, volume=46, number=1, pages=91–120, year=1995, publisher=Annual Reviews 4139 El Camino Way, PO Box 10139, Palo Alto, CA 94303-0139, USA


[33] Mihalcea, Rada and Corley, Courtney and Strapparava, Carlo and others: Corpus-based and knowledge-based measures of text semantic similarity. AAAI, pages=775–780, 2006

[34] Dagan, Id and Glickman, Oren: Probabilistic textual entailment: Generic applied modeling of language variability, year=2004

[35] Chowdhury, Gobinda G: Natural language processing, journal=Annual review of information science and technology, volume=37, number=1, pages=51–89, year=2003, publisher=Wiley Online Library


