Lexical Disambiguation of Igbo using Diacritic Restoration

Conference Paper · April 2017

CITATIONS
0

READS
17

3 authors:

Ignatius Majesty Ezeani
The University of Sheffield
12 PUBLICATIONS **23** CITATIONS
SEE PROFILE

M. Hepple
The University of Sheffield
99 PUBLICATIONS **1,496** CITATIONS
SEE PROFILE

Ikechukwu Onyenwe
9 PUBLICATIONS **8** CITATIONS
SEE PROFILE

Lexical Disambiguation of Igbo through Diacritic Restoration

Ignatius Ezeani **Mark Hepple** **Ikechukwu Onyenwe**
NLP Research Group,
Department of Computer Science,
University of Sheffield, UK.
{ignatius.ezeani,m.r.hepple,i.onyenwe}@sheffield.ac.uk

Abstract

Properly written texts in Igbo, a low resource African language, are rich in both orthographic and tonal diacritics. Diacritics are essential in capturing the distinctions in pronunciation and meaning of words, as well as in lexical disambiguation. Unfortunately, most electronic texts in diacritic languages are written without diacritics. This makes diacritic restoration a necessary step in corpus building and language processing tasks for languages with diacritics. In our previous work, we built some $n-$gram models with simple smoothing techniques based on a closed-world assumption. However, as a classification task, diacritic restoration is well suited for and will be more generalisable with machine learning. This paper, therefore, presents a more standard approach to dealing with the task which involves the application of machine learning algorithms.

Is diacritic restoration really a classification task??

1 Introduction

Diacritics are marks placed over, under, or through a letter in some languages to indicate a different sound value from the same letter. English does not have diacritics (apart from a few borrowed words) but many of the world's languages use a wide range of diacritized letters in their orthography. Automatic Diacritic Restoration Systems (ADRS) enable the restoration of missing diacritics in texts. Many forms of such tools have been proposed, designed and developed but work on Igbo is still in its early stages.

1.1 Diacritics and Igbo language

Igbo, a major Nigerian language and the native language of the people of the south-eastern Nigeria, is spoken by over 30 million people worldwide. It uses the Latin scripts and has many dialects. Most written works, however, use the official orthography produced by the *Ọnwụ Committee*[1].

The orthography has 8 vowels (*a, e, i, o, u, ị, ọ, ụ*) and 28 consonants (*b, gb, ch, d, f, g, gw, gh, h, j, k, kw, kp, l, m, n, nw, ny, ṅ, p, r, s, sh, t, v, w, y, z*).

Table 1, shows Igbo characters with their orthographic or tonal (or both) diacritics and possible changes in meanings of the words they appear in[2].

Char	Ortho	Tonal
a	–	à,á, ā
e	–	è,é, ē
i	ị	ì, í, ī, ị, ́ı, ı̄
o	ọ	ò, ó, ō, ọ, ọ́, ọ̄
u	ụ	ù, ú, ū, ụ, ụ́, ụ̄
m	–	m̀,ḿ, m̄
n	ṅ	ǹ,ń, n̄

Table 1: Igbo diacritic complexity

Most Igbo electronic texts collected from social media platforms are riddled with flaws ranging from dialectal variations and spelling errors to lack of diacritics. For instance, consider this raw excerpt from a chat on a popular Nigerian online chat forum *www.nairaland.com*[3]:

> *otu ubochi ka'm no na amaghi ihe mu na uwa ga-eje. kam noo n'eche ihe a,otu mmadu wee kpoturum,m lee anya o buru nwoke mara mma puru iche,mma ya turu m n'obi.o gwam si nne kedu*

[1] http://www.columbia.edu/itc/mealac/pritchett/00fwp/igbo/txt_onwu_1961.pdf

[2] *m* and *n*, nasal consonants, are sometimes treated as tone marked vowels.

[3] Source: *http://www.nairaland.com/189374/igbo-love-messages*

Proceedings of the 1st Workshop on Sense, Concept and Entity Representations and their Applications, pages 53–60,
Valencia, Spain, April 4 2017.

k'idi.onu ya dika onu ndi m'ozi, ihu ya dika anyanwu ututu,ahu ya n'achakwa bara bara ka mmiri si n'okwute. ka ihe niile no n'agbam n'obi,o sim na ohuru m n'anya.na ochoro k'anyi buru enyi,a hukwuru m ya n'anya.anyi wee kwekorita wee buru enyi onye'a m n'ekwu maka ya bu odinobi m,onye ihe ya n'amasi m

In the above example, you can observe that there is zero presence of diacritics - tonal or orthographic - in the entire text. As pointed out above, although there are other issues with regards to standard in the text, lack of diacritics seems to be harder to control or avoid than the others. This is partly because diacritics or lack of it does affect human understanding a great deal; and also the rigours a writer will go through to insert them may not worth the effort. The challenge, however, is that NLP systems built and trained with such poor quality non standard data will most likely be unreliable.

1.2 Diacritic restoration and other NLP systems

Diacritic restoration is important for other NLP systems such as speech recognition, text generation and machine translations systems. For example, although most translation systems are now very impressive, not a lot of them support Igbo language. However, for the few that do (e.g. *Google Translate*), diacritic restoration still plays a huge role in how well they perform. The example below shows the effect of diacritic marks on the output of *Google Translate*'s Igbo-to-English translation.

Statement	*Google Translate*	Comment
O ji *egbe* ya gbuo *egbe*	He used his **gun** to kill *gun*	wrong
O ji **égbè** ya gbuo **égbé**	He used his **gun** to kill **kite**	correct
Akwa ya di n'elu *akwa* ya	It was on the **bed** in his room	fair
Ákwà ya di n'elu **àkwà** ya	his **clothes** on his **bed**	correct
Oke riri *oke* ya	Her addiction	confused
Òké riri **òkè** ya	**Mouse** ate his **share**	correct
O jiri *ugbo* ya bia	He came with his *farm*	wrong
O jiri **ụgbọ** ya bia	He came with his **car**	correct

Table 2: Diacritic disambiguation for *Google Translate*

1.3 Diacritic restoration and WSD

Yarowsky (1994a) observed that, although diacritic restoration is not a hugely popular task in NLP research, it shares similar properties with such tasks as word sense disambiguation with regards to resolving both syntactic and semantic ambiguities. Indeed it was referred to as an instance of a closely related class of problems which includes word choice selection in machine translation, homograph and homophone disambiguation and capitalisation restoration (Yarowsky, 1994b).

Diacritic restoration, like sense disambiguation, is not an end in itself but an *"intermediate task"* (Wilks and Stevenson, 1996) which supports better understanding and representation of meanings in human-machine interactions. In most non-diacritic languages, sense disambiguation systems can directly support such tasks as machine translation, information retrieval, text processing, speech processing etc. (Ide and Véronis, 1998). But it takes more for diacritic languages, where possible, to produce standard texts. So for those languages, to achieve good results with such systems as listed above, diacritic restoration is required as a boost for the sense disambiguation task.

Flatly DISAGREE

What about electronic human to human interactions?

We note however, that although diacritic restoration is related to word sense disambiguation (WSD), it does not eliminate the need for sense disambiguation. For example, if the wordkey *akwa* is successfully restored to *àkwà*, it could still be referring to either *bed* or *bridge*. Another good example is the behaviour of *Google Translate* as the context around the word *àkwà* changes.

Statement	*Google Translate*	Comment
Akwa ya di n'elu *akwa*	It was on the high	confused
Akwa ya di n'elu *akwa* ya	It was on the bed in his room	fair
Ákwà ya di n'elu **àkwà**	His clothing was on the bridge	okay
Ákwà ya di n'elu **àkwà** ya	His clothing on his bed	good

Table 3: Disambiguation challenge for *Google Translate*

The last two statements, with proper diacritics on the ambiguous wordkey *akwa* seem both correct. Some disambiguation system in Google Translate must have been used to select the right form. However, it highlights the fact that such a disambiguation system may perform better when diacritics are restored.

2 Problem Definition

As explained above, lack of diacritics can often lead to some lexical ambiguities in written Igbo sentences. Although a human reader can, in most cases, infer the intended meaning from context, the machine may not. Consider the sentences in sections 2.1 and 2.2 and their literal translations:

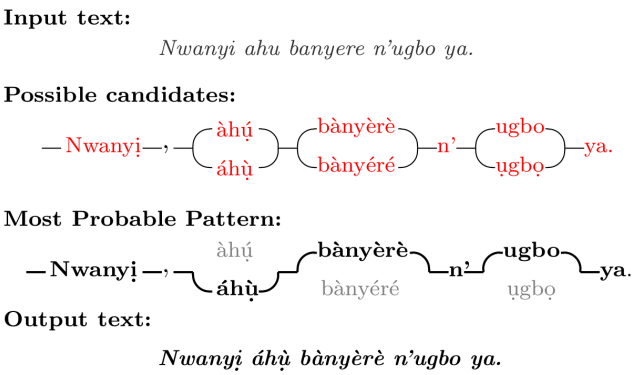

Figure 1: Illustrative View of the Diacritic Restoration Process (Ezeani et al., 2016)

2.1 Missing orthographic diacritics

1. *Nwanyi ahu banyere n'**ugbo** ya.* (The woman entered her **[farm|boat/craft]**)
2. *O kwuru banyere **olu** ya.* (He/she talked about his/her **[neck/voice|work/job]**)

2.2 Missing tonal diacritics

1. *Nwoke ahu nwere **egbe** n'ulo ya.* (That man has a **[gun|kite]** in his house)
2. *O dina n'elu **akwa**.* (He/she is lying on the **[cloth|bed,bridge|egg|cry]**)
3. ***Egwu** ji ya aka.* (He/she is held/gripped by **[fear|song/dance/music]**)

Ambiguities arise when diacritics – orthographic or tonal – are omitted in Igbo texts. In the first examples, we could see that **ugbo**(*farm*) and **ụgbọ**(*boat/craft*) as well as **olu**(*neck/voice*) and **ọlụ**(*work/job*) were candidates in their sentences.

Also the second examples show that **égbé**(*kite*) and **égbè**(gun); **ákwà**(cloth), **àkwà**(bed or bridge), **àkwá**(egg), or even **ákwá**(cry) in a philosophical or artistic sense; as well as **égwù**(fear) and **égwú**(music) are all qualified to replace the ambiguous word in their respective sentences.

3 Related Literature

Diacritic restoration techniques for low resource languages adopt two main approaches: *word based* and *character based*.

3.1 Word level diacritic restoration

Different schemes of the word-based approach have been described. They generally involve *preprocessing*, *candidate generation* and *disambiguation*. Simard (1998) applied POS-tags and HMM language models for French. On the Croatian language, Šantić et al. (2009) used substitution schemes, a dictionary and language models in implementing a similar architecture. For Spanish, Yarowsky (1999) used dictionaries with decision lists, Bayesian classification and Viterbi decoding the surrounding context.

Crandall (2005), using Bayesian approach, HMM and a hybrid of both, as well as different evaluation method, attempted to improve on Yarowsky's work. Cocks and Keegan (2011) worked on Māori using naïve Bayes and word-based *n*-grams relating to the target word as instance features. Tufiş and Chiţu (1999) used POS tagging to restore Romanian texts but backed off to character-based approach to deal with "unknown words". Generally, there seems to be a consensus on the superiority of the word-based approach for well resourced languages.

3.2 Grapheme or letter level diacritic restoration

For low-resource languages, there is often lack of adequate data and resources (large corpora, dictionaries, POS-taggers etc.). Mihalcea (2002) as well as Mihalcea and Nastase (2002) argued that letter-based approach will help to resolve the issue of lack of resources. They implemented instance based and decision tree classifiers which gave a high letter-level accuracy. However, their evaluation method implied a possibly much lower word-level accuracy.

Versions of Mihalcea's approach with improved evaluation methods have been implemented on other low resourced languages (Wagacha et al., 2006; De Pauw et al., 2011; Scannell, 2011). Wagacha et al. (2006), for example, reviewed the evaluation method in Mihalcea's work and introduced a word-level method for Gĩkũyũ. De Pauw et al. (2011) extended Wagacha's work by applying the method to multiple languages.

Our earlier work on Igbo diacritic restoration (Ezeani et al., 2016) was more of a proof of concept aimed at extending the initial work done by Scannell (2011). We built a number of n–gram models – basically unigrams, bigrams and trigrams – along with simple smoothing techniques. Although we got relatively high results, our evaluation method was based on a closed-world assumption where we trained and tested on the same set of data. Obviously, that assumption does not

model the real world and so it is being addressed in this paper.

3.3 Igbo Diacritic Restoration

Igbo is low-resourced and is generally neglected in NLP research. However, an attempt at restoring Igbo diacritics was reported by Scannell (2011) in which a combination of word- and character-level models were applied. Two lexicon lookup methods were used: *LL* which replaces ambiguous words with the most frequent word and *LL2* that uses a bigram model to determine the right replacement.

They reported word-level accuracies of 88.6% and 89.5% for the models respectively. But the size of training corpus (31k tokens with 4.3k word types) was too little to be representative and there was no language speaker in the team to validate the data used and the results produced. Therefore, we implemented a range of more complex n-gram models, using similar evaluation techniques, on a comparatively larger sized corpus (1.07m with 14.4k unique tokens) and had improved on their results (Ezeani et al., 2016).

In this work, we introduce machine learning approaches to further generalise the process and to better learn the intricate patterns in the data that will help better restoration.

4 Experimental Setup

4.1 Experimental Data

The corpus used in these experiments were collected from the Igbo version of the bible available from the Jehova Witness website[4]. The basic corpus statistics are presented in Table 4.

In Table 4, we refer to the "latinized" form of a word as its *wordkey*[5]. Less than 10% (529/15696) of the wordkeys are ambiguous. However, these ambiguous wordkeys represent 529 ambiguous sets that yield 348,509 of the corpus words (i.e. words that share the same *wordkey* with at least one other word). These ambiguous words constitutes approximately 38.22% (348,509/911892) of the entire corpus. Some of the top most occurring, as well as the bottom least occurring ambiguous sets are shown in Table 5.

[4] jw.org

[5] Expectedly, many Igbo words are the same with their wordkeys

Item	Number
Total tokens	1070429
Total words	902150
Numbers/punctuations	168279
Unique words	563383
Ambiguous words	348509
Wordkeys	15696
Unique wordkeys	15167
Ambiguous wordkeys	529
2 variants	502
3 variants	15
4 variants	10
5 variants	2
>5 variants	0
Approx. ambiguity	38.22%

Table 4: Corpus statistics

Top	Variants(count)
na(29272)	ná(1332), na(27940)
o(22418)	o(4757), ò(64), ó(5), ọ(17592)
Bottom	**Variants(count)**
Giteyim(2)	Giteyịm(1), Giteyim(1)
Galim(2)	Galim(1), Galịm(1)

Table 5: Most and least frequent wordkeys

4.2 Preprocessing

The preprocessing task relied substantially on the approaches used by Onyenwe et al. (2014). Observed language based patterns were preserved. For example, *ga–*, *na–* and *n'* are retained as they are due to the specific lexical functions the special characters "**–**" or " **'** " confer on them. For instance, while *na* implies conjunction (e.g. **ji *na* ede**: yam and cocoa-yam), *na–* is a verb auxiliary (e.g. **Obi *na–*agba ọsọ**: Obi is running) and *n'* is shortened form of the preposition *na* (e.g. **Ọ dị *n'*elu àkwà**: It is on the bed.).

Also for consistency, diacritic formats are normalized using the unicode's *Normalization Form Canonical* NFC composition. For example, the character *é* from the combined unicode characters *e* (u0065) and ´ (u0301) will be decomposed and recombined as a single canonically equivalent character *é* (u00e9). Also the character *ṅ*, which is often wrongly replaced with *ñ* and *n̄* in some text, is generally restored back to its standard form.

The diacritic marking of the corpus used in this research is sufficient but not full or perfect. The orthographic diacritics (mostly dot-belows) have

been included throughout. However, the tonal diacritics are fairly sparse, having been included only where they were for disambiguation (i.e. where the reader might not be able to decide the correct form from context.

Therefore, through manual inspection, some observed errors and anomalies were corrected by language speakers. For example, 3153 out of 3154 occurrences of the key *mmadu* were of the class *mmadụ*. The only one that was *mmadu* was corrected to *mmadụ* after inspection. By repeating this process, a lot of the generated ambiguous sets were resolved and removed from the list to reduce the noise. Examples are as shown in the table below:

wordkey	Freq	var1Freq	var2Freq
akpu	106	ákpụ–1	akpụ–105
agbu	112	agbụ–111	ágbụ–1
aka	3690	aka–3689	ákà–1
iri	2036	iri–2035	ịrị–1

Table 6: Some examples of corrected and removed ambiguous set

4.3 Feature extraction for training instances

The feature sets for the classification models were based on the works of Scannell (2011) on character-based restoration which was extended by Cocks and Keegan (2011) to deal with word-based restoration for Māori. These features consist of a combination of n-grams – represented in the form *(x,y)*, where x is the relative position to the target key and y is the token length – at different positions within the left and right context of the target word. The datasets are built as described below for each of the ambiguous keys:

- **FS1[(-1,1), (1,1)]:** Unigrams on each side of the target key
- **FS2[(-2,2), (2,2)]:** Bigrams on each side
- **FS3[(-3,3), (3,3)]:** Trigrams on each side
- **FS4[(-4,4), (4,4)]:** 4-grams on each side
- **FS5[(-5,5), (5,5)]:** 5-grams on each side
- **FS6[(-2,1), (-1,1), (1,1), (2,1)]:** 2 unigrams on both sides
- **FS7[(-3,1), (-2,1), (-1,1), (1,1), (2,1), (3,1)]:** 3 unigrams on each side
- **FS8[(-4,1), (-3,1), (-2,1), (-1,1), (1,1), (2,1), (3,1), (4,1)]:** 4 unigrams on each side
- **FS9[(-5,1), (-4,1), (-3,1), (-2,1), (-1,1), (1,1), (2,1), (3,1), (4,1), (5,1)]:** 5 unigrams on each side
- **FS10[(-2,2), (-1,1), (1,1), (2,2)]:** 1 unigram and 1 bigram on each side
- **FS11[(-3,3), (-2,2), (2,2), (3,3)]:** 1 bigram and 1 trigram on each side
- **FS12[(-3,3), (-2,2), (-1,1), (1,1), (2,2), (3,3)]:** 1 unigram, 1 bigram and 1 trigram on each side
- **FS13[(-4,4), (-3,3), (-2,2), (-1,1), (1,1), (2,2), (3,3), (4,4)]:** 1 unigram, 1 bigram, 1 trigram and a 4-gram on each side

4.3.1 Appearance threshold and stratification

We removed low-frequency wordkeys in our data by defining an *appearance threshold* as a percentage of the total tokens in our data. This is given by the

$$appThreshold = \frac{C(wordkeys)}{C(tokens)} * 100$$

and wordkeys with *appThreshold* below the stated value[6] were removed from the experiment.

As part of our data preparation for a standard cross-validation, we also passed each of our datasets through a simple stratification process. Instances of each label[7], where possible, are evenly distributed to appear at least once in each fold or removed from the dataset.

Our stratification algorithm basically picks only labels from each dataset that have a population p such that $p >= nfolds$. $nfolds$ is the number of folds which in our case has a default value of 10. In order to make the task a little more challenging, this process was augmented by the removal of some high frequency, but low *entropy* datasets where using the most common class (MCC) produces very high accuracies[8]. Entropy is loosely used here to refer to the degree of dominance of a particular class across the dataset and it is simply defined as:

$$entropy = 1 - \frac{max[Count(label_i)]}{len(dataset)}$$

[6] In this work, we used an $appThreshold$ of 0.005%

[7] $label$s are basically diacritic variants.

[8] Datasets with more than 95% accuracy on the most common class (i.e. with entropy lower than 0.05) were removed.

where $i = 1..n$ and n =number of distinct labels in the dataset. Table 7 shows 10 of the 30 lowest entropy datasets that were removed by this process.

wordkey	Counts	MCCscore	Label(count)
e(2)	5476	99.78%	è(12); e(5464)
anyi(2)	5390	99.63%	anyị(5370); ànyị(20)
ma(2)	6713	99.61%	ma(6687); mà(26)
ike(2)	3244	99.54%	ikè(15); ike(3229)
unu(2)	8662	99.53%	ùnu(41); unu(8621)
Ha(2)	2266	99.29%	Hà(16); Ha(2250)
a(2)	12275	99.10%	a(12165); à(110)
onye(2)	8937	98.87%	onye(8836); ònye(101)
ohu(2)	790	98.73%	ohu(780); ọhụ(10)
eze(2)	2633	98.14%	eze(2584); ezé(49)

Table 7: Low entropy datasets

At end of these pruning processes, our remaining datasets came to 110 with the distribution as follows:

- *datasets with only 2 variants:* = 93
- *datasets with 3 variants:* = 7
- *datasets with 4 variants:* = 8
- *datasets with 5 variants:* = 2

Some datasets that originally had multiple variants lost some of their variants. For example, the dataset from *akwa* which originally had five variants and 1067 instances comprising of *ákwá* (355), *ákwà*(485), *akwa*(216), *àkwà*(1) and *àkwá*(10) retained only four variants (after dropping *àkwà*) and 1066 instances.

4.4 Classification algorithms

This work applied versions of five of the commonly used machine learning algorithms in NLP classification tasks namely:

- Linear Discriminant Analysis(LDA)
- K Nearest Neighbors(KNN)
- Decision Trees(DTC)
- Support Vector Machines(SVC)
- Naïve Bayes(MNB)

Their default parameters on Scikit-learn toolkit were used with 10-fold cross-validation and the evaluation metrics used is mainly the accuracy of prediction of the correct diacritic form in the test data. The effect of the accuracy obtained for a dataset on the overall performance depends on the weight of the dataset. Each dataset is assigned a weight corresponding to the number of instances it generates from the corpus which is determined by its frequency of occurrence.

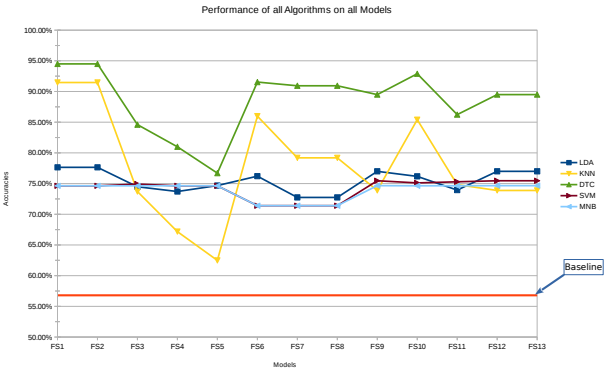

Figure 2: Evaluation of algorithm performance on each feature set model

So the actual performance of each learning algorithm, on a particular feature set model, is the overall weighted average of the its performances across all the 110 datasets. The *bottom line* accuracy is the result of replacing each word with its $wordkey$ which gave an accuracy of 30.46%. However, the actual baseline to beat is **52.79%** which is achieved by always predicting the most common class.

4.5 Results and Discussions

The results of our experiments are as shown in Table 8 and Figure 2.

Models	LDA	KNN	DTC	SVM	MNB
Baseline:	**52.79%**				
FS1	**77.65%**	**91.47%**	**94.49%**	74.64%	74.64%
FS2	**77.65%**	**91.47%**	**94.49%**	74.64%	74.64%
FS3	74.48%	73.70%	84.60%	74.92%	74.64%
FS4	73.71%	67.18%	81.00%	74.64%	74.64%
FS5	74.68%	62.48%	76.70%	74.64%	74.64%
FS6	76.21%	85.98%	91.54%	71.39%	71.39%
FS7	72.74%	79.20%	90.94%	71.39%	71.39%
FS8	72.74%	79.20%	90.94%	71.39%	71.39%
FS9	76.99%	73.88%	89.50%	**75.46%**	**74.67%**
FS10	76.18%	85.41%	92.89%	75.11%	74.64%
FS11	73.94%	74.83%	86.23%	75.29%	74.64%
FS12	76.99%	73.88%	89.50%	**75.46%**	**74.67%**
FS13	76.99%	73.88%	89.50%	**75.46%**	**74.67%**

Table 8: Summary of results

The experiments indicate that on the average all the algorithms were able to beat the baseline on all models. The decision tree algorithm (DTC) performed best across all models with an average accuracy of 88.64% (Figure 3), and the highest accuracy of 94.49% (Table 8) on both the **FS1** and **FS2**

models. However, with an average standard deviation of 0.076 (Figure 4) for its results, it appears to be the least reliable.

As the next best performing algorithm, KNN falls below DTC in average accuracy (91.47%) but seems slightly more reliable. It did, however, struggle more than others as the dimension of feature n-grams increased (see its performance on **FS3**, **FS4** and **FS5**). This may be due to the increase in sparsity of features and the difficulty to find similar *neighbours*. The other algorithms – **LDA**, **SVM** and **MNB** – just trailed behind and although their results are a lot more reliable especially **SVM** and **MNB** (Figure 4). But this may be an indication that their strategies are not explorative enough. However, it could be observed that they traced a similar path in the graph and also had their highest results with the same set of models (i.e. **FS9**, **FS12** and **FS13**) with wider context.

On the models, we observed that the unigrams and bigrams have better predictive capacity than the other n-grams. Most of the algorithms got comparatively good results with **FS1**, **FS2**, **FS6** and **FS10** (Figure 5) each of which has the unigram closest to the target word (i.e. in the ± 1 position) in the feature set. Also, models that excluded the closest unigrams on both sides (e.g. **FS11**) and those with fairly wider context did not perform comparatively well across algorithms.

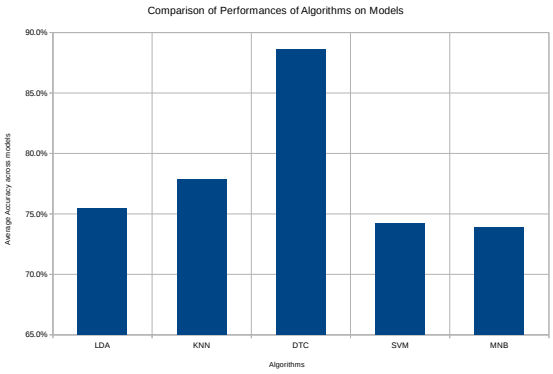

Figure 3: Average performance of algorithms

Again, it appears that beyond the three closest unigrams (i.e. those in the -3 through $+3$ positions), the classifiers tend to be confused by additional context information. Generally, **FS1** and **FS2** stood out across all algorithms as the best models while **FS6** and **FS7** also did well especially with **DTC**, **KNN** and **LDA**.

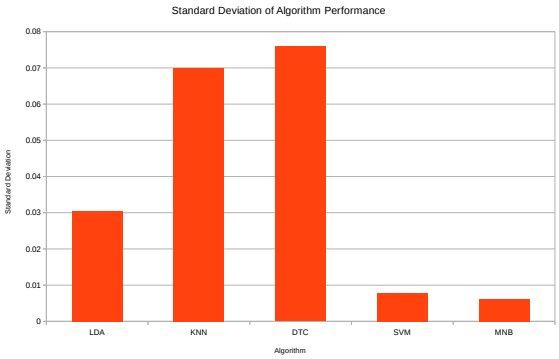

Figure 4: Average standard deviation for algorithms

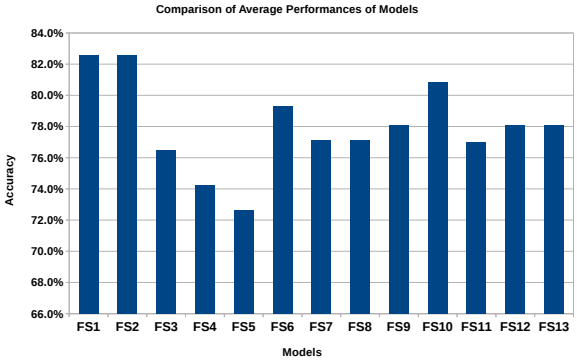

Figure 5: Average performance of models

4.6 Future Research Direction

Although our results show a substantial improvement from the baseline accuracy by all the algorithms on all the models, there is still a lot of room for improvement. Our next experiments will involve attempts to improve the results by focusing on the following key aspects:

- *Reviewing the feature set models:*
  So far we have used instances with similar features on both sides of the target words. In our next experiments, we may consider varying these features.

- *Exploiting the algorithms:*
  We were more explorative with the algorithms and so only the default parameters of the algorithms on Scikit-learn were tested. Subsequent experiments will involve tuning the parameters of the algorithms and possibly using more evaluation metrics.

- *Expanding data size and genre:*
  A major challenge for this research work is lack of substantially marked corpora. So although, we achieved a lot with the bible data,

OMG Yes! it is inadequate and not very representative of the contemporary use of the language. Future research efforts will apply more resources to increasing the data size across other genres.

- *Predicting unknown words:*
  Our work is yet to properly address the problem of unknown words. We are considering a closer inspection of the structural patterns in the target word to see if they contain elements with predictive capacity.

- *Broad based standardization:*
  Beside lack of diacritics online Igbo texts are riddled with spelling errors, lack of standard orthographic and dialectal forms, poor writing styles, foreign words and so on. It may therefore be good to consider a broader based process that includes, not just diacritic restoration but other aspects of standardization.

IOHAVOC

- *Interfacing with other NLP systems:*
  Although it seems obvious, it will be interesting to investigate, in empirical terms, the relationship between diacritic restoration and others NLP tasks and systems such as POS-tagging, morphological analysis and even the broader field of word sense disambiguation.

Acknowledgements

The IgboNLP Project @ The University of Sheffield, UK is funded by TETFund & Nnamdi Azikiwe University, Nigeria.

References

John Cocks and Te-Taka Keegan. 2011. A Word-based Approach for Diacritic Restoration in Māori. In *Proceedings of the Australasian Language Technology Association Workshop 2011*, pages 126–130, Canberra, Australia, December.

David Crandall. 2005. Automatic Accent Restoration in Spanish text. http://www.cs.indiana.edu/∼djcran/projects/674_final.pdf. [Online: accessed 7-January-2016].

Guy De Pauw, Gilles maurice De Schryver, Laurette Pretorius, and Lori Levin. 2011. Introduction to the Special Issue on African Language Technology. *Language Resources and Evaluation*, 45:263–269.

Ignatius Ezeani, Mark Hepple, and Ikechukwu Onyenwe, 2016. *Automatic Restoration of Diacritics for Igbo Language*, pages 198–205. Springer International Publishing, Cham.

Nancy Ide and Jean Véronis. 1998. Introduction to the special issue on word sense disambiguation: The state of the art. *Comput. Linguist.*, 24(1):2–40, March.

Rada F. Mihalcea and Vivi Nastase. 2002. Letter level learning for language independent diacritics restoration. *In: Proceedings of CoNLL-2002, Taipei, Taiwan*, pages 105–111.

Rada F. Mihalcea. 2002. Diacritics Restoration: Learning from Letters versus Learning from Words. *In: Gelbukh, A. (ed.) CICLing LNCS*, pages 339–348.

Ikechukwu Onyenwe, Chinedu Uchechukwu, and Mark Hepple. 2014. Part-of-speech Tagset and Corpus Development for igbo, an African Language. *LAW VIII - The 8th Linguistic Annotation Workshop.*, pages 93–98.

Kevin P. Scannell. 2011. Statistical unicodification of african languages. *Language Resource Evaluation*, 45(3):375–386, September.

Michel Simard. 1998. Automatic Insertion of Accents in French texts. *Proceedings of the Third Conference on Empirical Methods in Natural Language Processing*, pages 27–35.

Dan Tufiş and Adrian Chiţu. 1999. Automatic Diacritics Insertion in Romanian texts. In *Proceedings of COMPLEX99 International Conference on Computational Lexicography*, pages 185–194, Pecs, Hungary.

Nikola Šantić, Jan Šnajder, and Bojana Dalbelo Bašić, 2009. *Automatic Diacritics Restoration in Croatian Texts*, pages 126–130. Dept of Info Sci, Faculty of Humanities and Social Sciences, University of Zagreb , 2009. ISBN: 978-953-175-355-5.

Peter W. Wagacha, Guy De Pauw, and Pauline W. Githinji. 2006. A Grapheme-based Approach to Accent Restoration in Gĩkũyũ. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation*, pages 1937–1940, Genoa, Italy, May.

Yorick Wilks and Mark Stevenson. 1996. The grammar of sense: Is word-sense tagging much more than part-of-speech tagging? *CoRR*, cmp-lg/9607028.

David Yarowsky. 1994a. A Comparison of Corpus-based Techniques for Restoring Accents in Spanish and French Text. In *Proceedings, 2nd Annual Workshop on Very Large Corpora*, pages 19–32, Kyoto.

David Yarowsky. 1994b. Decision Lists for Lexical Ambiguity Resolution: Application to Accent Resolution in Spanish and French Text. In *Proceedings of ACL-94*, Las Cruces, New Mexico.

David Yarowsky, 1999. *Corpus-based Techniques for Restoring Accents in Spanish and French Text*, pages 99–120. Kluwer Academic Publishers.