Processing Low-Resource Languages: A case of the Yoruba Language

Author: Idris Abdulhameed Oyebode

Supervisor: Dr. Mark-Jan Nederhof

University of
St Andrews

Co-funded by the
Erasmus+ Programme
of the European Union

A dissertation submitted in partial fulfilment of the requirements for the Double Erasmus Mundus MSc in Advanced Systems Dependability (DEPEND)

June 2019

DECLARATION

I declare that the material submitted for assessment is my own work except where credit is explicitly given to others by citation or acknowledgement. This work was performed during the current academic year except where otherwise stated.

The main text of this project report is NN,NNN* words long, including project specification and plan.

In submitting this project report to the University of St Andrews, I give permission for it to be made available for use in accordance with the regulations of the University Library. I also give permission for the title and abstract to be published and for copies of the report to be made and supplied at cost to any bonafide library or research worker, and to be made available on the World Wide Web. I retain the copyright in this work.

ACKNOWLEDGEMENTS

I would like to offer special thanks and appreciation to my supervisor, Dr. Mark-Jan Nederhof of the School of Computer Science at the University of St. Andrews for his tremendous, valuable and unwavering help and support throughout the stages of this project. I would also like to thank my advisor, Dr. Juliana Bowles of the same School, who has been a fantastic advisor.

ABSTRACT

There are an estimated 6-7 thousand languages in the world, of which a large chunk of them are low-resource languages. As of May 2017, Google's multilingual translator, Google Translate, supports only a little over 100 languages which is a very small percentage of the total languages present. This, among other reasons has resulted in more researches into developing natural language processing tools for many of these languages and to improve the existing ones. However, many of these are low-resource languages and are morphologically rich, therefore often lack the annotated resources needed to develop excellent natural language processing tools for them. In this project, we present, analyze and compare multiple statistical approaches to performing automatic morphosyntactic annotation of texts in the Yoruba Language.

Based on the small tagged corpus available for the Yoruba Language, we performed supervised training by tagging using a baseline model, a Hidden Markov Model and Cross-Lingual transfer learning model that uses 9 different resource-rich languages as source languages. Also, an unsupervised learning model was trained for application on a much larger and unannotated corpus. In our evaluation, the models performed very well even with the problem of sparse data. We also discovered that the cross-lingual model performed a little better than the baseline model and a naïve transition probability model. English, an S-V-O language like Yoruba, performed best in this aspect. Danish and Portuguese, which are both Indo-European Languages were the second-best performers. Naija Pidgin, which has its root in Nigeria completed the top 4.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

The world is a linguistically diverse place. In Africa alone, it is estimated that there are between 1500 to 2000 languages which represents over 20% of the languages in the world. A lot of these languages are only spoken but not written as half of the world's languages is estimated to only be spoken (Lewis, 2009). Language consist of many structures, which includes sounds, words, morphology, part of speech, syntax, semantics and discourse. In the production and understanding of language, humans easily and almost effortlessly assimilate all these structures. The objective of Natural Language Processing is to achieve at least this also, albeit with a computer. The greatest impediment to statistical processing of most of these languages has been the lack of large enough monolingual and/or parallel corpora with adequate linguistic resources for developing Natural Language Processing applications. The languages that fall under this category are called Low-Resource Languages. Other categories are the Resource-Rich Languages (includes English, French, Spanish. Arabic, Chinese etc.) and the Medium-Resource Languages (includes Czech, Hindi, Hebrew among others).

Some of the core technologies needed to perform this task are Language modelling, Part-of-Speech tagging, syntactic parsing, among others. The applications of these technologies range from Information Retrieval, Question Answering, Machine Translation and many more. For this thesis, we experiment and investigate some of the known statistical approaches to Part-of-Speech tagging for Low-Resource Languages, in this case, the Yoruba Language. To do this, we need annotated datasets with a trusted and consistent schema. The annotated scheme in the Universal Dependencies (UD) project which was founded on the progression of universal Stanford dependencies (de Marneffe et al., 2006, 2008, 2014), Google Universal Part-of-Speech tags (Petrov et al., 2012) and the Interset interlingua for morphosyntactic tagsets (Zeman, 2008) are most suitable for this. "Universal Dependencies is a project that seeks to develop cross-linguistically consistent treebank annotation for many languages, with the goal of facilitating multilingual parser development, cross-lingual learning, and parsing research from a language typology perspective." (Nivre et al., 2018)

Chapter 1 of this thesis discusses the Yoruba language structure including its' part-of-speech and morphology. Chapter 2 gives a context survey and description of past and recent related works on part-of-speech tagging in low-resource and resource rich languages.

Chapter 3 describes our work- the baseline model, the supervised learning with Hidden Markov Models, cross-lingual transfer learning using 9 languages and the unsupervised learning with expectation-maximization (EM) algorithm. Chapter 4 reports the intrinsic evaluation of each of these models and draws comparison amongst them and with respect to resource rich languages such as the English language.

Objectives the Project

The primary objectives of the project are:

- Bootstrapping a part-of-speech tagger from the limited available annotated data
- Training a Hidden Markov Model with a training data
- Decoding (Testing) the Hidden Markov Model on a test data

Secondary objectives include:

- Implementing Cross-lingual transfer learning with resource rich languages as source languages.
- Training a Hidden Markov Model using unsupervised learning.

1. THE YORUBA LANGUAGE

1.1 Background

The Yoruba language is one of the Benue-Congo subclasses of the Niger-Congo class/family of languages (Adeniyi, 2007). It is a predominantly spoken language in West Africa with over 40 million native speakers. Nigeria and Republic of Benin have the majority of its speakers; however, it is also spoken in other places around the world such as Trinidad and Tobago, Brazil, Cuba and some parts of Europe (Fabunmi, 2005). Yoruba is a 'dialect continuum' more than 12 dialects which includes Ègbá, Òyó, Òwó, Ìjèbú, Ìjèsà among others. Each of these dialects vary in their phonologies, lexicons and even grammar (Bamgbose, 1966; UCLA).

1.2 Standard Yoruba

Of all the dialects of the Yoruba language, there is a dialect which is understood and spoken across cultures. It is called Yoruba Ajumolo (Standard Yoruba). This is the dialect that is used to teach and taught in educational institutions, it is also the one used in other formal sectors like government agencies, the social and mass media. This dialect is the one adopted for this project and used in this dissertation.

1.3 Orthography of Yoruba

Initially, Yoruba was written in a form of Arabic script which coincides with the claim that the Yoruba ethnic group has its roots from the Arabian Peninsula. "The Modern Yoruba Orthography which used the Latin alphabet originated in the early work of the Church Mission Society (CMS) by Bishop Ajayi Crowther. Early versions of the bible were translated by Bishop Ajayi Crowther using the Latin alphabet but without enough tone markings. The only diacritic used was the under dot." (Oluokun, 2018).

There are 3 phonological components of the standard Yoruba language. They are tones, vowels and consonants. There are 3 tones namely: do (low), mi (medium) and re (high) popularly called doremi. The tones appear only on vowels and syllabic nasal. Each syllable carries a tone indicated by their respective tonal signs. There are 18 consonants and twelve vowels of which 7 are oral verbs and 5 are nasal. The nasal verbs are- *an, on, in, on* and *un*. Tables 1.1 and 1.2 below show the consonants and oral vowels of Yoruba, their example usage in words and meanings.

S/N	Consonants	Words	Meanings
1	b	bo	peel
2	d	de	arrive
3	f	fe	want

4	g	ga	tall
5	gb	gba	take
6	h	ha	tight
7	j	je	eat
8	k	ka	pluck
9	l	lu	beat
10	m	mu	drink
11	n	nu	feed
12	p	pa	kill
13	r	ra	buy
14	s	se	did
15	s	sha	fade
16	t	ti	close
17	w	wa	drive
18	y	ya	borrow

Table 1. 1 Yoruba Consonants

S/N	Verbs	Words	Meanings
1	a	ka	pluck
2	e	ke	cry
3	e	ke	pamper
4	i	ki	praise
5	o	ko	pack

6	o	ko	build
7	u	ku	died

Table 1. 2 Yoruba oral vowels

1.4 Part of Speech

The most notable and well-known parts of speech in Yoruba are Nouns, Pronouns and Verbs. Other parts of speech which are less notable are adjectives, adverbs and prepositions. Bamgbose (2010) divided these parts of speech into multiple classes based on their behaviors or functions. Nouns in Yoruba have multiple categories which include countable and uncountable nouns, concrete and abstract nouns, human and non-human nouns, descriptive nouns, among others. Verbs, like nouns also have multiple classes, they include narrative verbs, adjectival verbs, repetitive verbs and so on. Pronouns can be subject, object or possessive and subject to their cases, gender and person. Adverbs are either sentence-based or noun qualifiers.

1.5 Morphology

Morphology of a language involves the ways in which words are formed in that language. The words are formed from some units called morphemes which are either stems (core units) or affixes (bits that stick to the stem). Morphology mostly can be inflectional, compounding, derivational or clitisational. Most of the Yoruba morphology is derivational. This means that to an extent, lemmatization is less complicated in the language which is a feature of languages that are inflectional. Yoruba words are formed in three major ways- compounding, affixation and reduplication.

1.5.1 Compounding

A lot of Nouns in the Yoruba language are formed through compounding. Sometimes, this involves deletion of some characters when forming a new word, other times, the whole characters of the words combining to form a new word are retained. Some examples are shown below:

1) gbale + gbale = gbalegbale
   sweep + sweep = cleaner
2) oni + eko = oluko
   owner of + education = lecturer
3) a + gba + ejo + ro = agbejoro
   the + take + discussion + discuss = lawyer

From the above examples, the first one involves combining two words without deleting any character. The second involves the characters o, n, and I combining to become “lu” and the third involves deletion of a character “a” in gba.

1.5.2 Affixation

Adewole (1995) and Awoyale (1981) described the process of affixation in Yoruba language using prefixes and their various classes. This is one of the most predominant ways of word formation in Yoruba. Most of the time, the prefixes are attached to predictive phrases. Some examples of these are listed below:

1) i + to = ito
   PREFIX + to take care = saliva
2) i + ja = ija
   PREFIX + to fight = fight
3) i + gba = igba
   PREFIX + take = time
4) e + to = eto
   PREFIX + to line up = program

1.5.3 Reduplication

This method of word formation is also common in Yoruba and continues to expand with different generations. It involves repetition of a word that was formed usually by compounding. Folarin (1989) classified the resulting words from this repetition into 3 groups - quantified nouns, agentive nouns from phrasal verbs and adverbs of time/manner. Some examples of this are shown below:

1) da + oju = dajudaju
   clear + eye = certainly
2) egbe + ki + egbe = egbekegbe
   group + ki + group = bad company
3) eni + ki + eni =enikeni
   person + ki + person = anyone
4) gbe + omo = gbomogbomo
   carry + child = kidnapper

2. RELATED LITERATURE

This chapter presents the works that have been done by past and recent researchers in Part-of-speech tagging using different approaches on resource rich languages and low resource languages. The most common models that have been used over the years have been with Hidden Markov Models, Maximum Entropy Models and more recently Neural Networks.

In the mid-90s, rule-based/logic-based models were the standards for part of speech tagging and NLP in general. They consist of two stages (Klein and Simmons, 1963) which were assigning words with lists of their potential parts of speech using a dictionary. These lists are sorted down in the second stage using some disambiguation rules. Of course, there were setbacks with this approach as it becomes more difficult to build rules for large datasets and they do not generalize well. Therefore, they had a very low recall although with high precision.

In the late 20th century to the early 2000s, stochastic taggers were introduced for part-of-speech tagging (Church, 1988; DeRose, 1988). Most notable among them are the Hidden Markov Models (HMM) and Maximum Entropy Models (MEMM) both of which are variants of Markov's models. They are more robust in the face of real-world data unlike rule-based taggers and therefore have better performance. They also require less engineering of hand-crafted rules. However, they do not perform optimally on low-resource languages due to the limitations of these languages being low-resource.

In recent times, approximately since the mid-2010s, statistical Neural taggers, otherwise called Neural Networks have had significant improvement in the processing of Natural languages in general. Like HMMs, they also have better performance and more robustness. But statistical Neural taggers generally do not perform too well in low-resource settings and may over generalize. They also require very large datasets to perform very well even on resource rich languages. In general, the state-of-the-art NLP approaches require very large training data and usually some feature engineering on the specific language to perform excellently. This is expensive as it requires linguistically trained experts in the language domain. Also, as we have noted earlier, this is an even greater challenge when it comes to low-resource languages.

2.1 Part-of-Speech Tagging on Resource-Rich languages

Resource-rich languages such as English have been the earliest means of developing part-of-speech taggers. This has led to the creation of famous corpora like the 1-million-word Brown corpus which had 87 tags (Francis and Kucera, 1982). DeRose (1998) mentions that only 11.5% of the word types in this corpus are ambiguous, however, 40% of its tokens are ambiguous. Another famous corpus is the Penn Treebank with 45 tags (Marcus et al., 1993). These two are widely used for POS tagging in English till today. Marcus et. al (1993) also discovered that only 96-97% of the tags are agreed on by human annotators, this means the statistical taggers generally aren't expected to go beyond that upper bound. This is referred to as the Gold standard in tagging.

Different approaches have been used to perform POS tagging on these corpuses which has achieved varying results. Gale et al. (1992) proposed the standard baseline for tagging which involves tagging each word with its most likely tag. This is called the baseline because it is expected to be the infimum of any POS tagger. This simple baseline tagger achieved about 90-91% accuracy on the Brown corpus (Charniak et al., 1993). The use of Hidden Markov Models for tagging was fully introduced after this, even though there had been some applications of pseudo-HMMS in previous years by Church (1988, 1989). However, he had incorrectly used the probability of a tag given a word $P\ (tag|word)$ instead of $P\ (word|tag)$ as his emission probability which he claimed was more understandable as "storing a lexicon in an almost standard form". With HMMs, accuracies above 95% have been achieved depending on how much augmentation is done and Maximum Entropy Models (MEMMs) have achieved approximately 97% which is the Gold Standard.

In recent times, neural networks, especially recurrent networks have been used for POS tagging. They have achieved tremendous results especially for very large training sets. Ling et al. (2015) achieved 95.93 and 97.36% on the Penn Treebank using Bi-RNN and Bi-LSTM models which involved learning word representations through character embeddings. At present, the highest state of the art model achieved 97.96% on the Penn Treebank by Bohnet et al. (2018) in which they combined word encodings and initial character with meta-BiLSTM.

2.2 Part-of-Speech Tagging on Resource-Poor languages

One of the earliest and most successful work in POS tagging for low-resource languages through projection was by Yarowsky et al. (2001) wherein a POS tagger was made to learn from a dictionary and a reference grammar. They explored the possibility of projecting linguistic annotations such as POS tags from one language to another. They experimented using supervised learning with English as source language and French and Chinese as target languages by using training techniques enhanced for noisy data. With this, they achieved an accuracy in the range of around 94-96%.

HMMs, which are sequence and generative models have been used to characterize stochastic dependencies between successive tags. Li et al. (2012), Das and Petrov (2011) and Tackstrom et al. (2013) all used HMMs for weakly supervised learning from token or type constraints. Li et al (2012) used a general-purpose dictionary called tag Wiktionary wherein they labelled data to train and evaluate with eight different European languages and achieved high results which were state-of-art. Although it could be argued that these European languages are no longer low resource at present, but they almost were low resource as at when the research was carried out.

Das and Petrov (2011) presented an unsupervised approach to induce part-of-speech taggers for low-resource languages which have no tags associated with their training data although they have translations in a parallel resource rich language. They projected annotations with high confidence using a graph-based approach from the resource rich language which is the source language to the resource poor language which are subsequently used for unsupervised learning. The Expectation-Maximization (EM) algorithm is used to induce the parameters of a trigram Hidden Markov Model. With this, they achieved a 16.7% accuracy over the ordinary HMMs. Their approach only differs slightly from Yarowsky

and Ngai (2001) in that they used a graph-based approach and they trained the resulting tag set using unsupervised training rather than supervised. A drawback on this and why it isn't applicable to our project is that it requires a large parallel data for training, which of course, we do not have.

Tackstrom et al. (2013) further built on the work done by Li et al. (2012) by exploring the results of projecting token and type constraints on different models. They used a partially observed Conditional Random Field (CRF) which achieved a 25% increase in accuracy over Li et al's (2012) state of the art, averaged over only the 8 Indo-European languages earlier used. They achieved an 88.8% average over all the 8 languages except for Turkish which achieved 65.2%. However, Turkish, like the Yoruba language, is a morphologically rich language. Also, Tackstrom et al used 15 languages in their own project.

Sukhareva et al. (2017) investigated the performance of POS taggers on Hittite language which is low resource using distant supervision because there are no available morpho syntactic annotations for it. They also used POS projections from German translations of a Hittite transcript through character-based alignments and applied stemming to solve the problem of data sparsity. With this they achieved a 69% accuracy. A drawback to this also just like Das and Petrov (2011) is that a large parallel data for projection is needed.

Garrete et al. (2013) used semi-supervised learning to experiment POS tagging on two low resource languages namely; Kinyarwada and Malagasy. They showed that annotating word types is the most important aspect of the process as far as enough type information can always be projected onto a raw corpus through the semi-supervised learning model. They also used Finite State Transducers (FSTs) for morphological analysis which produced sets of morphological features for word types.

Buys and Botha (2016) proposed a "discriminative embedding-based model with ranked learning" to train POS taggers for resource poor languages having rich tag sets. This was done without direct supervision and evaluated on 11 languages. They discovered that the model performs best when projections are done between languages that are related, and achieves an accuracy on par with a simple weakly supervised HMM. The model was also an extension over the Wsabie model of Weston et al. (2011) which is a "shallow Neural Network that learns to optimize precision at the top of a ranked list of label." We discuss more on neural networks for POS tagging in the next section.

2.3 Part-of-Speech tagging for Low-Resource Languages using Neural Networks

Neural Networks have been the dominant algorithms in recent times for many statistical applications involving data. However, there hasn't been much work done on low-resource languages using Neural Networks. This is because Neural Networks perform best on very large datasets for training which of course, is the drawback with low-resource languages. However, there very few which have been achieved with Recurrent Neural Networks and bi-LSTM. Plank and Agic (2018) explored some of the methods earlier mentioned such as tag dictionaries (Li et al., 2012) and annotation projection with type constraints (Yarowsky, et al., 2001; Das and Petrov, 2011; Tackstrom et al., 2013) combined with other methods as morphological lexicons and instance selection all in a single and uniform structure to

develop a POS tagger that scales to multiple low-resource languages using distance supervision from disparate sources. They experimented on the 8 languages used by Das and Petrov (2011) and achieved 86.2% accuracy against the state of the art's 87.3% on average, but their model outperformed that of Das and Petrov in French, Italian , Czech and Spanish. However, based on their evaluation on 25 languages in total, they achieved a state of the art by recording substantial gains on many of them.

Zennaki et al. (2015) had earlier developed a multilingual POS tagger using a Recurrent Neural Network (Mikolov et al., 2010) based only on word occurrence in a parallel corpus and a POS tagger in the source language. Like Petrov et al. (2012) and Yarowsky et al. (2001), they projected linguistic annotations from the resource rich language, although this time, the information accompanied with word alignment was not used. They evaluated across English, Greek, German, French and Spanish, from which they achieved a performance close to the state of the art. Their result also showed that adapting the unsupervised RNN performs well in low-resource settings.

3. PART-OF-SPEECH TAGGING

Natural Language Processing is a task that is rife with ambiguity problems at different levels (word sense, part of speech, syntax, et cetera). These ambiguities of course are beneficial as they give us multiple options in understanding concepts, however, they need to be resolved, else they become problems. Resolving these ambiguities is to a large extent the whole essence of Natural Language Processing models, which is to choose the correct analysis in different contexts.

Part-of-speech tagging involves a "process of assigning a part-of-speech or other syntactic class marker to each word in a corpus" (Jurafsky and Martin; 2008). To do this, a POS tagging model is needed which takes as input a string of words, known as observations, and produces as output, the parts of speech of each of those words. This, of course looks like a trivial task, however, as earlier mentioned, the problem with this arises with ambiguities. For example, the word *"bank"* in English language has more than one part-of-speech tag, depending on its context, it could be a noun or a verb, therefore it is said to be ambiguous. The objective of a POS model tagger then, is to resolve this and many more ambiguities common in languages through a process called disambiguation. This, we explored using different approaches.

3.1 Universal Dependencies Part-of-Speech tags

The treebanks in the Universal Dependencies (UD) project (de Marneffe et al., 2014) from which we extracted the corpora used in this project largely have a set of 17 tags which is an extension of the 12 tags proposed by Petrov et al. (2012). We used UD v2.3 (Nivre et al., 2018) which has 76 languages annotated with morphological features. The 17 tags along with their meanings are listed in table 3.1 below:

POS tags	Meaning
ADP	Adposition
ADJ	Adjective
AUX	Auxiliary
ADV	Adverb
CCONJ	Coordinating Conjunction
DET	Determiner
INTJ	Interjection
NOUN	Noun

NUM	Numeral
PART	Particle
PROPN	Proper Noun
PRON	Pronoun
PUNCT	Punctuation
SCONJ	Subordinating Conjunction
SYM	Symbol
VERB	Verb
X	Other

Table 3. 1 Universal Dependencies (UD) POS tags

3.2 The Data

As mentioned above, the data for the project was extracted from the UD treebank. The Yoruba corpus, which is our main concern created by Oluokun et al. (2018) based on POS annotation projections idea of Yarowsky et al. (2001). This corpus however has 15 tags of the universal tags' 17, it does not contain SYM and INTJ tags as at the v2.3 of the treebank used for this project. For this reason, for our cross-lingual model, we explored with only treebanks of languages with 15 tags in the UD treebank. The Yoruba corpus, unlike most of the resource-rich languages in the UD treebank has only one treebank. This comprises of 2,664 tokens/words and 100 sentences, which obviously, is a small corpus. We divided this corpus into training and test sets in the 75:25 ratio. We obviously could not use a held out/development set because the of the size of our data. And it is for this reason that we couldn't have applied deep learning algorithms also, i.e. Neural Networks.

Other data used in the project- specifically for cross-lingual transfer learning, were not split into training and test set, because they are the source languages from which their transition probabilities will be transferred/projected onto the Yoruba language. So, the whole dataset for each of them was used. Finally, modern diachronic variations and the rich morphology of the Yoruba language exacerbated by its tonal attributes mentioned in chapter 1 further increases the data sparsity which further makes the statistical tagging uneasy. However, it equally improves the possibility of easy disambiguation by our statistical models.

3.3 POS Tagging

3.3.1 The Baseline

The baseline as suggested by Gale et al. (1992) is the process of tagging each word in a sentence/observation with its most frequent/likely tag. This is the simplest case of Markov models called the unigram model of POS tagging. It is, as opposed to the Gold standard- which is the human ceiling, expected to be the floor value for tagging. Here, only the emission probabilities of each word with respect to a tag was computed. This follows the unigram model of not using any history in computation. Mathematically, it is as follows:

$$P(tag_n \mid word_1, word_2 \ldots word_n) \approx P(tag_n \mid word_n)$$

This indicates that the probability of a word having a tag relies only on the conditional probability of the word with the tag. This, we did by computing the joint probability of the tag, and the word as a ratio of the probability of the word.

$$P(tag \mid word) = \frac{P(word, tag)}{P(word)}$$

Therefore, the $P(tag \mid word)$ was computed for every word observed in the input over all the 15 tags and the tag with the highest probability is selected as the POS tag for that word. So, what about unknown words? That is, when a word appears in our test set and we have not seen it before, then this means the $P(tag \mid word)$ for that word will be equal to zero (0) and therefore generate the tag "X" which represents a tag for a probability of a word that is unknown. We could decide to ignore this, since it does not affect the next or previous word. However, there are different approaches to this, to combat this for the baseline method, we tagged the unknown words with the parts of speech with the highest frequencies and also arbitrary tagging to see how it performs.

3.3.2 Supervised POS Tagging with Hidden Markov Models

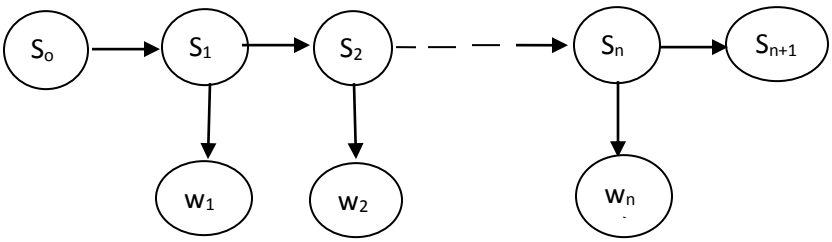
3.3.2.1 Hidden Markov Models

Having achieved a baseline tagger which implements a unigram model, we proceeded with a more intelligent approach which implements a bigram model. HMMs are generative sequence classifiers that assign labels or classes to units of a sequence (Jurafsky and Martin, 2008). It is one of the most important models in machine learning for language processing and has been the classical solution for POS tagging for over two decades. Hidden Markov Models are an extension of Markov chains and Finite State Transducers. With it, we can compute the most likely tag for a word while considering a short history of the tag and the unigram probability of the word with respect to the tag.

Formally, in terms of a Finite Automata, our HMM, just like HMMs in general, is defined as a 5-tuple specified by the following components:

- $S = s_1 s_2 s_3 \ldots s_N$ defined as a set of N states.
- $O = o_1 o_2 o_3 \ldots o_T$ defined as a set of T output symbols/observations drawn from a vocabulary, Σ.

- $A == a_{11}a_{12}a_{13}\ldots a_{n1}\ldots a_{nm}$ defined as the transition probability matrix, with each a_{ij} representing the probability of moving from a state, *i* to another state, *j,* subject to $\sum_{j=1}^{n} aij$ for all *i.*
- $B = b_i(o_t)$ defined as a sequence of observation likelihoods called emission probabilities which is the probability of the observation o_t being generated in state*, i*.
- s_0, s_F, defined as the initial/start state and final state which are elements of the set of states.



$$P(s,w) = \left(\prod_{k=1}^{n} P\,(w_k \mid s_k)\, P(s_k \mid s_{k-1})\right)$$

Figure 3. 1 Hidden Markov models with states, s_t and observations, w_t at each time step

However, we generaly do not usually refer to the initial and final states exclusively in this project. Therefore, when we say transition probabilities, we imply that the start and end state probabilities are also covered therein.

3.3.2.2 HMM Training

HMMs are a special case of Bayesian Inference (Bayes, 1763) popularly known as Bayes rule defined as:

$$P(x|y) = \frac{P(y|x)\,P(x)}{P(y)}$$

Where *x and y* are events. Our desire is to find the probabilities of having the correct tags for each word in a sequence of words, this, estimated as:

$$\hat{t}_1^n = \underset{\hat{t}_1^n}{argmax}\, P(\hat{t}_1^n \mid w_1^n)$$

Combining these two above equations gives,

$$\hat{t}_1^n = \underset{\hat{t}_1^n}{argmax} \frac{P(w_1^n \mid t_1^n)\, P(t_1^n)}{P(w_1^n)}$$

Where w_1^n and t_1^n represent the words and tags from 1 to n. The $\underset{\hat{t}_1^n}{argmax}$ means we are trying to maximize the probability of getting the correct tags. By normalization, the denominator is removed, since it doesn't change for each tag sequence, that is, it is a constant. So that we have,

$$\hat{t}_1^n = \underset{\hat{t}_1^n}{argmax}\, P(w_1^n \mid t_1^n)\, P(t_1^n)$$

Now, the two terms left are called the prior probability $P(t_1^n)$ of the tag sequence and the likelihood $P(w_1^n \mid t_1^n)$ of the word string. This model above is sometimes called the "noisy channel model". In more formal terms, the two terms are called POS transition and emission probabilities respectively. HMMs make two assumptions to simplify this, as even with this equation, it is still too hard to compute. This is a feature of first order HMMs. Firstly, the emission probability is assumed as the probability of a word is dependent only on its part of speech and not of any other word or tags around it. Secondly, for the transition probability, it is assumed that the probability of a tag is dependent only on the tag that appears before it. This is where our bigram model comes in. The two assumptions are represented mathematically as:

$$P(w_1^n \mid t_1^n) \approx \prod_{k=1}^{n} P(w_k \mid t_k)$$

And

$$P(t_1^n) \approx \prod_{k=1}^{n} P(t_k \mid t_{k-1})$$

However, for our model, we considered the starting and ending probabilities for the transition probabilities, therefore, editing the transition probability to become,

$$P(t_1^n) \approx \left(\prod_{k=1}^{n} P(t_k \mid t_{k-1)}) \right) P(\langle /s \rangle \mid t_n)$$

Where t_{-0} is the start-of-sentence tag represented as $\langle s \rangle$. Also, $\langle /s \rangle$ is the end-of-sentence tag. Lumping all these assumptions together gives us the most probable tag sequence as follows:

$$\hat{t}_1^n \approx \underset{\hat{t}_1^n}{argmax} \left(\prod_{k=1}^{n} P(w_1^n \mid t_1^n) P(t_k \mid t_{k-1)}) \right) P(\langle /s \rangle \mid t_n)$$

So, how do we obtain these conditional probabilities which are our transition and emission probabilities, which is otherwise known as our training/learning process? We compute the relative frequency estimate of our model probabilities using the training set by counting the number of occurrences as a ratio of their contexts. Mathematically, this means:

$$P(w_k \mid t_k) = \frac{count\ (t_k, w_k)}{count\ (t_k)}$$

And

$$P(t_k \mid t_{k-1}) = \frac{count\ (t_{k-1}t_k)}{count\ (t_{k-1})}$$

3.3.3 Cross-lingual Transfer Learning with HMMs

Transfer learning has been the most frequently used approach to low-resource Natural Language Processing for both semi-supervised and unsupervised learning (Tackstrom et al. 2013, Das and Petrov 2011, Yarowsky and Ngai 2001, Duong et al. 2013, Hwa et al. 2005, Ma and Xia 2014). It is useful when there is a very small amount of annotated data or there is no annotated data at all. Many natural languages have been found to share similar properties. This includes parts of speech, word order, syntax, among others. This is the motivation behind transfer learning. It involves transfer of resources or models from rich resource language to low resource languages. In other words, Transfer leaning is "motivated by the fact that people can intelligently apply knowledge learned previously to solve new problems faster or with better solutions" and therefore "allows the domains, tasks, and distributions used in training and testing to be different" (Pan et al., 2010) . Its variations are called Zero-shot learning and One-shot learning. Generally, a cross-lingual transfer learning task can either involve:

- Transfer of models: that is training a model in a resource rich language and applying it in a resource poor language.
- Transfer of annotations: annotations could include part of speech tags, syntactic features, word orders among others which could be transferred via cross-lingual bridges such as word or phrase alignments.

For most low-resource NLP tasks where there isn't annotated data at all, then the transfer of annotations is usually first done which includes word alignments, in which parts of speech from the resource rich language is projected to the resource poor language. However, for our project, we had annotations already for our small annotated corpus from the UD treebank, therefore, we transfer only our transition probabilities model from resource rich languages to the Yoruba language. This means we are essentially transferring the syntax and word order from the resource rich languages to the Yoruba language. The emission probabilities from the small Yoruba corpus is used here. So, we have a model with transition probabilities, $P(t_k \mid t_{k-1})$ from the resource rich languages and the emission probabilities, $P(w_k \mid t_k)$ from Yoruba language. With this, the languages that have the highest accuracies are those most similar to Yoruba in terms of word order and to some extent, syntax. The mathematical representation of this is only slightly different from our initial supervised tagging such that;

$$\hat{t}_1^n \approx \underset{\hat{t}_1^n}{argmax} \left(\prod_{k=1}^{n} P_{RR}(w_1^n \mid t_1^n)\, P_{RP}(t_k \mid t_{k-1}) \right) P(\langle /s \rangle \mid t_n)$$

Where P_{RR} means transition probabilities of resource rich language and P_{RP} means emission probabilities of resource poor language.

Given some languages in the UD treebank have the standard 17 tags while some others have 15 tags, it becomes difficult to appropriately perform transfer learning with some of the resource rich languages as there will be a tag mismatch, so, we used only some of those languages with 15 tags for our experiment. The resource rich languages used are English, Italian, Spanish, Catalan, French, Naija, Swedish, Portuguese and Danish. Smoothing was also applied on both the transition and emission probabilities in this case using Witten-Bell smoothing. This will be explained in subsequent chapters.

3.3.4 Unsupervised POS Tagging with HMMs

Tagging of parts of speech without supervision has been a very essential part of Natural Language Processing for resource poor languages. With it, things like annotation cost reduction can be achieved, that is, resource poor languages without any annotated data can be roughly annotated and which can then be subsequently refined by human annotators. Hidden Markov Models are still the go-to models for this task till today as they perform considerably well. With unsupervised training, we can learn the parameters of our HMM, that is the transition and emission probabilities. This approach to POS tagging is also called part-of-speech induction. Generally, for this algorithm to be effective, a lot of data is needed which we do not have because we are working on low resource.

The task here is to learn the parameters of the HMM given/subject to an observation sequence, that is, raw sentences, a set of states and the parameters in the HMM. To do this, we use the standard unsupervised message passing algorithm for HMMs called the Baum-Welch algorithm (Baum, 1972; Welch, 2003). This algorithm is an instance/special case of a class of algorithm called Expectation-Maximization (EM) algorithms (Dempster et al., 1977). This is started with a usually uniform HMM to which one does not have priori access to the distributions. However, since we have a small annotated corpus for Yoruba already, we decided to generate our transition and emission probabilities of the HMM using this corpus. This is the same way we did for our supervised learning approach. This implies that, even though the task is unsupervised, in practice, setting the initial conditions for the HMM is very important as it determines which local optimum the unsupervised training may end up in. So, this is seen as an attempt to give the algorithm more information for optimum performance.

Thereafter, we trained these parameters using the Baum-Welch algorithm. This algorithm uses two intuitions; firstly, iteratively estimating counts for an observation sequence. This involves estimating the transition and emission probabilities, then deriving better probabilities from them. The second intuition involves getting the forward probability of an observation and dividing the probability mass among all

the different paths that contributed to this forward probability. This is then used to get our estimated probabilities (Jurafsky and Martin, 2008). With this, the algorithm finds the maximum likelihood estimate of the parameters of our HMM, given the observations. Learning using this approach however does not guarantee arriving at a global optimum as it is highly non-convex and likely to get stuck in local optima (Johnson, 2007).

The expectation step of the Baum-Welch algorithm uses forward and backward passes to compute the probability of seeing some observation $w_1 w_2 w_3 \ldots w_i$ at a particular point in time. The maximization step updates our transition and emission probabilities using weighted counts (the forward and backward probabilities derived). In implementing this algorithm, we separate our initial probabilities from the transition probabilities for better understanding, however, not doing so will still get correct results. The algorithm makes the process clearer as shown below:

Set parameters of *HMM = (A, B)*

function BAUM-WELCH (*observations* of len *T*, *emission probabilities V, hidden states S*) **returns** *A, B*

Initialize A and B

Iterate until convergence

E-step

$$\alpha_k(s) = P(w_0\, w_1 \ldots\ w_k, s_k)$$

$$= \sum_{s_{k-1}} P(s_k \mid s_{k-1})\, P(w_k \mid s_k)\, \alpha_{i-1}(s_{k-1})$$

$$\beta_k(s) = P(w_{k+1} \ldots\ w_n \mid s_k)$$

$$= \sum_{s_{k+1}} P(s_{k+1} \mid s_k)\, P(w_{k+1} \mid s_{k+1})\, \beta_{i+1}(s_{k+1})$$

M-step

$$\gamma_k(s) = \frac{\alpha_k(s)\ \beta_k(s)}{\sum_{j=1}^{N} \alpha_j(s)\ \beta_j(s)}$$

$$\xi_{kj}(s) = \frac{\alpha_k(s)\, a_{kj}\, \beta_j\,(s+1)\, b_j\,(w_{k+1})}{\sum_{k=1}^{N} \sum_{j=1}^{N} \alpha_k(s)\, a_{kj}\, \beta_j\,(s+1)\, b_j\,(w_{k+1})}$$

The parameters of the HMM are now updated by normalization as:

$$\hat{a}_{kj} = \frac{\sum_{s=1}^{S-1} \xi_{kj}(s)}{\sum_{s=1}^{S-1} \sum_{m=1}^{N} \xi_{km}(s)}$$

$$\hat{b}_k(v_m) = \frac{\sum_{k=1\ s.t. w_k = v_m}^{S-1} \gamma_k(s)}{\sum_{s=1}^{S} \gamma_k(s)}$$

return A, B

Figure 3. 2 The Baum-Welch Algorithm

$\alpha_k(s)$ represents the probability of seeing observations $w_1 w_2 w_3 \ldots w_k$ and being in state *k* at time *s,* this is called the forward probability. $\beta_k(s)$ is the probability of the ending partial sequence $w_{k+1} \ldots\ w_n$ given the start state *k* and time *s,* this is called the backward probability, which is a dual of the forward probability. $\gamma_k(s)$ is the expected emission probability count while $\xi_{kj}(s)$ is the expected transition

probability counts. However, the denominators of both are the same, they represent the probability of seeing an observation given the HMM parameters. $\hat{a}_{kj}$ and $\hat{b}_k$ are the new transition and emission probabilities derived by normalizing the expected transition and emission probability counts.

3.3.5 Smoothing

In all our approaches using Hidden Markov models, from tagging with supervised training to cross-lingual tagging and finally tagging with unsupervised training, an unavoidable problem we face throughout is the problem of data sparsity. This is because we have insufficient data for the Yoruba language, which will result in having some events having zero probabilities. In other words, when we encounter an unknown word, or even possibly an unknown bigram sequence, the conditional probability for that will be zero, which may result in tagging subsequent words/observations wrongly. Generally, data sparsity is a common problem in speech and language processing as even in extreme cases where there's an extremely large data for training, during which parameters of the model can then be trained without smoothing, one thus expands the model by moving to n-grams of higher order to improve performance. This then makes the more complex and data sparsity issue occurs again (Chen and Goodman, 1998). So, it is widely believed that if data sparsity problem does not occur in a language modelling or POS tagging problem, then the model is extremely too simple. However, as much as we try to avoid too complex models, we also do not want models that are too simple such that they would not learn appropriately, so a balance is struck between these two extremes.

There are many smoothing algorithms that have been invented and tested over the years. They include Good Turing (Good, 1953), Witten-Bell smoothing (Witten and Bell, 1991), Jelinek-Mercer smoothing (Jelinek and Mercer, 1980) and many more. For our project, we used Witten-Bell smoothing which like some others, use the count of things that has been seen once to estimate the count of things that have not been seen at all. That is, using the frequency of events that occur once (called singleton) to estimate that of bigrams with zero counts. This is generally called the Good-Turing intuition. The remaining probability mass is then discounted, such that all probability estimates sum to 1. Mathematically, the Witten-Bell smoothing is represented as thus:

$$P_{WB}\left(w_k \mid w_{k-n+1}^{k-1}\right) = \lambda_{t_{k-n+1}^{k-1}} P_{ML}\left(w_k \mid w_{k-n+1}^{k-1}\right) + \left(1 - \lambda_{t_{k-n+1}^{k-1}}\right) P_{WB}\left(w_k \mid w_{k-n+2}^{k-1}\right)$$

This is a general case of the smoothing, for our case which is a bigram model, we use the higher order model (bigram in this case) if the bigram w_{k-n+1}^{k-1} was seen in the training data, otherwise, we back off to the unigram model. The $1 - \lambda_{t_{k-n+1}^{k-1}}$ represents the probability of recurring to the unigram model, that is, that a word not seen after w_{k-n+1}^{k-1} in the training data occurs after that history in the test data. We estimate this by the number of unique words that follow the history w_{k-n+1}^{k-1} in the training data. We apply this smoothing for our transition and emission probabilities. To compute the λs, the number of unique words that follow the history w_{k-n+1}^{k-1} is computed as:

$$N_{1+}\left(w_{k-n+1}^{k-1} \blacksquare\right) = \left|\{w_k : c\left(w_{k-n+1}^{k-1} w_k\right) > 0\}\right|$$

And λs are set such that:

$$1 - \lambda_{t_{k-n+1}^{k-1}} = \frac{N_{1+}\left(w_{k-n+1}^{k-1} \blacksquare\right)}{N_{1+}\left(w_{k-n+1}^{k-1} \blacksquare\right) + \sum_{w_k} c\left(w_{k-n+1}^{k}\right)}$$

To demonstrate smoothing example from our training corpus, we considered the bigram histories "ọlọ́run" and "nígbà". They occur 30 and 14 times in the corpus respectively.

- 7 different words follow "ọlọ́run" and is almost always followed by "sì" which followed it 18 times. Then there are 5 singletons (words that only followed it once). The bigram "ọlọ́run sì" means "God then…" or "Then, God…"
- 3 different words follow "nígbà" and the most frequent word is "tí" which occurred 9 times followed by "náá" which occurred 4 times.

Applying our lambda parameter formulas to this gives,

$$1 - \lambda_{\text{ọlọ́run}} = \frac{N_{1+}(\text{ọlọ́run}, \blacksquare)}{N_{1+}(\text{ọlọ́run}, \blacksquare) + \sum_{w_k} c\left(\text{ọlọ́run}, w_n\right)}$$

$$= \frac{7}{7 + 30} = 0.1892$$

$$1 - \lambda_{\text{nígbà}} = \frac{N_{1+}(\text{nígbà}, \blacksquare)}{N_{1+}(\text{nígbà}, \blacksquare) + \sum_{w_k} c\left(\text{nígbà}, w_n\right)}$$

$$= \frac{3}{3 + 14} = 0.1765$$

3.3.6 Decoding

Part of speech tagging in general is a machine learning classification problem; this is because the objective is to predict the correct tag for every word in the input statement. However, it is a multiclass classification problem because there are multiple parts of speech that are possible for each word/observation in the input unlike in a binary classification where only two classes are possible. So, our task is to find the most likely sequence of tags/states for a given observation/sentence. This is called decoding. This is also a disambiguation task as we attempt to resolve ambiguities for the words in the input sentence.

One problem that immediately arises here is the problem of trajectories, that is, there are too many possible states/tags sequences to list for each sentence. This gives a complexity in the order of $O(S^N)$, where S is part of speech tags and N are the words. A fast and naive approach to handle this is the beam search, which involves choosing a set of partial hypotheses of the whole trajectory. Here, a *K* is set which denotes the beam width. The beam width represents the number of top paths to be kept at each derivation step and the others, discarded. This gives us the benefit of fast computation and works well in

practice. However, given it is not necessarily an exhaustive search and therefore does not guarantee to give optimal answers except when *K* is the same as the number of parts of speech/classes, therefore, with the beam search, it becomes difficult to validate our model.

For this reason, we need a more memorized and iterative solution. So, we used the Viterbi algorithm, which is the classical decoding algorithm for HMMs. It is a generalized form of beam search which is common in speech and language processing. It is an application of dynamic programming for computing which computes the best path up to a position, *i* in the input, ending in state, *s.* The dynamic program for computing is:

$$\delta_k(s) = \max_{s_0 \ldots s_{k-1} s} P(s_0 \ldots s_{k-1} s, w_1 \ldots w_{i-1})$$

Which for HMMs becomes,

$$\delta_k(s) = \max_{s'} P(s \mid s') P(w \mid s') \ \delta_{k-1}(s')$$

We used back pointers to keep track of the probabilities at each derivation step used to construct our best path, so we have the equation;

$$\varphi_k(s) = \operatorname{argmax}_{s'} P(s \mid s') P(w \mid s') \ \delta_{k-1}(s')$$

The Viterbi algorithm is only slightly different from our forward probability algorithm in that, at each derivation, it uses only the path with the maximum probability, unlike the forward probability algorithm which computes the sum of all paths to that derivation. The Viterbi algorithm can be written in multiple ways, our viterbi algorithm for decoding of the models is shown below:

function VITERBI (*observations* of len *T*, *state-graph* of N) returns *best-path*

create a path probability matrix *viterbi[N,T]*

for each state *s* from 1 to N, do

Viterbi [s, 1] $\leftarrow a_{0,s} * b_s(o_1)$

backpointer [s, 1] $\leftarrow 0$

for each time step *t* from 2 to *T* do

for each state s from 1 to N do

viterbi [s,t] $\leftarrow \max_{s'=1} viterbi\ [s', t-1] * a_{s',s} * b_s(o_t)$

backpointer [s, t] $\leftarrow \arg\max_{s'=1} viterbi\ [s', t-1] * a_{s',s} * b_s(o_t)$

Viterbi $[q_F, T] \leftarrow \max_{s=1} viterbi\ [s, T] * a_{s,qF}$

backpointer [s, t] $\leftarrow \arg\max_{s=1} viterbi\ [s, T] * a_{s',qF}$

bestpath ← the path starting at state *backpointer* that follows *backpointer [ ]* to

states back in time

return *bestpath*

Figure 3. 3 Viterbi Algorithm for finding best path in a set of hidden states.

4. EVALUATION

Evaluation of models is a very important aspect of Machine Learning and Natural Language Processing, without it, we wouldn't be able to assess the performance of our models. Generally, for part-of-speech tagging task, the dataset is split into three- training, development and test sets. However, due to our small dataset (corpus), we split the dataset into two sets, excluding the development set. With this, we can train on our models using training set and perform intrinsic evaluation of the models using the test set. As mentioned in chapter 3, we split our dataset in the 75:25 split using 75% of the data for training and 25% for testing. A problem that occurs with using fixed training and test set is we do not know if the test set is large enough to be representative, usually, for situations like this, cross-validation approach which uses all the data for both training and testing is usually the best approach. For this reason, we used a 4-fold cross-validation approach to test our models to be sure it generalizes correctly.

Accuracy has been the standard way of evaluating tagging models. This involves taking the ratio of the number of words tagged correctly to the total number of words tagged. Correctness of each tag is ascertained by comparing the tag with the gold standard. The gold standard is the corpus which has been annotated by hand by humans and agreed on to a very large extent. Another way of evaluating tagging is through standard classification metrics known in Machine Learning. They include precision, recall, f1-score and using confusion matrices. We explored all the aforementioned metrics in evaluating the performance of our different taggers for the different tagging approaches we used in chapter 3.

Models generally perform better with richer features such as prefixes, suffices, word shapes, lowercase, capitalization, et cetera. For the Yoruba language, most of these things do not apply, so, we used only a feature of converting all words into lowercase. This gave us improved performance for all our models.

4.1 The Baseline Model

For this model, which is a relatively simple model, we used a fixed training set and test set. We also used this same split for the other models we trained. For this, the data set sizes, and the number of unknowns is shown below:

Data Set	Tokens	Unknown
Training	1,748	
Test	928	188 (20.26%)

Table 4. 1 Data Sizes

We tested the model on the test set using different tags to represent unknown words including the unknown tag itself (X), we recorded the tags with the top 5 accuracies for this as shown in table 4.2 below:

Part of Speech	Accuracy (%) (Lowercase)	Accuracy (%) (Sentence case)
VERB	80.71	78.44
PROPN	78.99	72.09
NOUN	77.26	75.10
ADV	76.72	74.57
X	74.03	71.88

Table 4. 2 Top 5 Accuracies for Baseline model.

Form the above table, we can see that the accuracy of the baseline model increases from 74.03% to 80.71% when all unknowns are tagged as VERB which is a significant improvement. Also, we could see that the model has higher accuracy with our richer feature of using all words as lowercase words. However, we cannot be too sure this above accuracy is representative of the whole dataset, so we performed cross validation, and with this, discovered new findings as shown in table 4.3 below:

Folds	Accuracy (%) (Lowercase)	Accuracy (%) (Sentence case)
Fold 1	76.31	76.31
Fold 2	83.99	88.35
Fold 3	83.85	87.11
Fold 4	78.76	79.36
Mean	**80.73**	**82.78**

Table 4. 3 Cross validation results on Baseline Model

The cross-validation approach changed everything we had discovered and assumed, part of which was that richer features such as using lower case words for all words which is a significant feature for higher accuracies in English, however, this has proven not to be true in this case, as making all words lowercase reduced the accuracy. This is based on using VERB as a tag for all unknowns which is the highest of all tags. Using the model with the highest accuracy as a case study, we explored how often a tag was predicted with respect to its correct tag using two confusion matrices as shown in figure 4.1 and 4.2

below. Figure 4.1 is the usual confusion matrix that shows the number of correct classifications and misclassifications for each tag with respect to other tags and figure 4.2 shows it as a percentage.

Tags/classes	ADJ	ADP	ADV	AUX	CCONJ	DET	NOUN	NUM	PART	PRON	PROPN	PUNCT	SCONJ	VERB
ADJ	3	0	0	0	0	0	0	0	0	0	0	0	0	6
ADP	0	47	0	3	0	0	0	0	0	0	0	0	0	3
ADV	0	0	16	1	0	0	0	0	0	0	0	0	2	26
AUX	0	1	1	49	0	0	0	0	0	0	0	0	2	4
CCONJ	0	0	1	0	35	0	0	0	0	0	0	0	0	0
DET	0	0	4	0	0	14	0	0	0	0	0	0	0	0
NOUN	0	1	0	0	0	0	24	0	0	0	0	0	0	30
NUM	0	0	0	0	0	0	0	2	0	0	0	0	0	2
PART	0	0	0	0	0	0	0	0	8	0	0	0	0	0
PRON	0	0	0	2	0	1	0	0	2	173	0	0	0	10
PROPN	0	0	0	0	0	0	0	0	0	0	0	0	0	46
PUNCT	0	0	0	0	0	0	0	0	0	0	0	169	0	3
SCONJ	0	0	2	0	0	0	0	0	0	0	0	0	64	2
VERB	0	6	8	4	0	0	0	0	1	5	0	0	0	145

Figure 4. 1 Confusion Matrix (a) for the Baseline Tagger

Tags/classes	ADP	ADV	AUX	CCONJ	NOUN	PRON	PROPN	PUNCT	SCONJ	VERB
ADP	5.1	0	0.3	0	0	0	0	0	0	0.3
ADV	5.1	1.7	0.1	0	0	0	0	0	0.2	2.8
AUX	0.1	0.1	5.3	0	0	0	0	0	0.2	0.4
CCONJ	0	0.1	0	3.8	0	0	0	0	0	0
NOUN	0.1	0	0	0	2.6	0	0	0	0	3.2
PRON	0	0	0.2	0	0	18.6	0	0	0	1.1
PROPN	0	0	0	0	0	0	0	0	0	5.0
PUNCT	0	0	0	0	0	0	0	18.2	0	0.3
SCONJ	0	0.2	0	0	0	0	0	0	6.9	0.2
VERB	0.6	0.9	0.4	0	0	0.5	0	0	0	15.6

Figure 4. 2 Confusion Matrix (b) for the Baseline Tagger showing errors (%)

The second confusion matrix shows the top 10 tags that occur more frequently in the test set, with their common errors as a percentage. For example, the matrix shows that a substitution of VERB for PROPN occurred for 5.0% of the words, which obviously accounts for our tagging of unknown words with verbs and the fact that the next best accuracy achieved was when we tagged all unknown words as proper nouns. The confusion matrices have given us a lot of information, to evaluate this model more, and for more details on the information presented by the matrices, we inspected the precisions, recalls and f1 scores for each of the classes/tags in table 4.4 below.

Tags	Precision	Recall	F1 score
ADJ	1.00	0.33	0.50
ADP	0.85	0.89	0.87
ADV	0.50	0.36	0.42
AUX	0.83	0.86	0.84
CCONJ	1.00	0.97	0.99
DET	0.93	0.78	0.85
NOUN	1.00	0.44	0.61
NUM	1.00	0.50	0.67
PART	0.73	1.00	0.84
PRON	0.97	0.92	0.95
PROPN	0.97	0.92	0.94
PUNCT	1.00	0.98	0.99
SCONJ	0.94	0.94	0.94
VERB	0.52	0.86	0.65

Table 4. 4 Precision, Recall and F1 score for the tags with Baseline model

From the above scores, we could see some classes did very well on their f1 scores, such as PUNCT and CCONJ with 99%, which shows the model classified almost all their instances correctly and classes such as ADV performing very poorly with 42% which can be seen from our confusion matrix (a) where so many adverbs were classified as verbs, more than even the ones that were classified as adverbs. Some

classes have very good precision, but low recall, and vice versa. An example is PART which has a very high recall but lower precision due to PRON being misclassified as PART. Generally, in Machine Learning problems, there is always a tradeoff between precision and recall which is a very well-known phenomenon.

4.2 Supervised POS Tagging with HMMs

Firstly, we evaluated our model by measuring the accuracy of the correct tags as measured in baseline model. We also explored the richer feature of lowercase words as against the normal case. With this, we have the following accuracies

	Accuracy (%) (Lowercase)	**Accuracy (%) (Sentence case)**
Supervised HMM	85.35	78.77

Table 4. 5 Accuracy of Hidden Markov Model tagger

However, this is when an arbitrarily fixed test and training sets were used. For clarity, like we did in the case of the baseline model, we explored cross-validation using the same folds used for the baseline model to inspect how well our HMM model generalizes, this also presented us with new findings as shown in table 4.6 below:

Folds	**Accuracy (%) (Lowercase)**	**Accuracy (%) (Sentence case)**
Fold 1	78.38	79.17
Fold 2	87.04	86.75
Fold 3	86.13	86.13
Fold 4	83.58	82.40
Mean	**83.78**	**84.64**

Table 4. 6 Cross validation results on Supervised HMM Model

Again, as in the baseline model, our expected richer feature turned out to make our model poorer, as it can be seen that the HMM performs better on average when in sentence case than when in lower case. Like earlier, we went further to explore the confusion matrix for this HMM model and subsequently the precisions, recalls and f1 score for the model as shown in figures 4.3 and 4.4, and table 4.7 below:

Tags/classes	ADJ	ADP	ADV	AUX	CCONJ	DET	NOUN	NUM	PART	PRON	PROPN	PUNCT	SCONJ	VERB
ADJ	3	0	0	0	0	0	3	0	0	0	0	0	0	2
ADP	0	49	0	3	0	0	0	0	0	0	0	0	0	1
ADV	0	2	25	2	0	1	11	0	0	0	1	0	2	1
AUX	0	1	1	50	0	0	0	0	0	0	0	0	0	5
CCONJ	0	0	1	0	35	0	0	0	0	0	0	0	0	0
DET	0	0	3	0	1	14	0	0	0	0	0	0	0	0
NOUN	0	4	0	0	0	0	47	0	0	0	0	0	0	4
NUM	0	1	0	0	0	0	2	1	0	0	0	0	0	2
PART	0	0	0	0	0	0	0	0	8	0	0	0	0	0
PRON	0	2	0	2	0	1	3	0	0	172	0	0	0	8
PROPN	0	1	0	0	0	0	7	0	0	2	35	0	0	1
PUNCT	0	0	1	0	0	0	2	0	0	0	0	169	0	0
SCONJ	0	0	4	9	0	0	0	0	0	0	0	0	54	1
VERB	0	8	3	7	0	0	11	0	0	10	0	0	0	130

Figure 4. 3 Confusion Matrix (a) for the HMM Tagger

Tags/classes	ADP	ADV	AUX	CCONJ	NOUN	PRON	PROPN	PUNCT	SCONJ	VERB
ADP	5.3	0	0.3	0	0	0	0	0	0	0.1
ADV	0.2	2.7	0.2	0	1.2	0.1	0	0	0.2	0.1
AUX	0.1	0.1	5.4	0	0	0	0	0	0	0.5
CCONJ	0	0.1	0	3.8	0	0	0	0	0	0
NOUN	0.4	0	0	0	5.1	0	0	0	0	0.4
PRON	0.2	0	0.2	0	0.3	18.5	0	0	0	0.9
PROPN	0.1	0	0	0	0.8	0	3.8	0	0	0.1
PUNCT	0	0.1	0	0	0.2	0	0	18.2	0	0
SCONJ	0	0.4	1.0	0	0	0	0	0	5.8	0.1
VERB	0.9	0.3	0.8	0	1.2	1.1	0	0	0	14.0

Figure 4. 4 Confusion Matrix (b) for the HMM Tagger showing errors (%)

Tags	Precision	Recall	F1 score
ADJ	1.00	0.33	0.50
ADP	0.72	0.92	0.81
ADV	0.66	0.56	0.60
AUX	0.68	0.88	0.77
CCONJ	0.97	0.97	0.97
DET	0.88	0.78	0.82
NOUN	0.55	0.85	0.67
NUM	1.00	0.25	0.40
PART	1.00	1.00	1.00
PRON	0.92	0.91	0.92
PROPN	1.00	0.76	0.86
PUNCT	1.00	0.98	0.99
SCONJ	0.96	0.79	0.87
VERB	0.85	0.77	0.81

Table 4. 7 Precision, Recall and F1 score for the tags with Supervised HMM model

From the above confusion matrices and table, it is easily noticeable why this model has a higher performance than our baseline model, although the baseline model has better f1 scores on some classes such as PROPN and SCONJ, the HMM model was generally more precise in correctly classifying the classes it sees better than the Baseline model, while competing very well also in their recalls. This is the reason that even though the baseline model had a high True Positive value for VERB with 145 and amounting to 15.6% of the words being correctly tagged as so, it still has a lower f1 score than the HMM model on that class. This is because the HMM model performs better in the trade-off between precision and recall and does not totally sacrifice one at the expense of the other.

4.3 Cross-lingual Transfer Learning with HMM

As we did for the other models, we measured the accuracy of the correct tags for this model for each of the different languages used as source language for transfer of transition parameters. With this, we got the following accuracies as shown in table 4.8.

	Accuracy (%)								
Source Languages	English	Catalan	Spanish	French	Swedish	Portuguese	Naija	Danish	Italian
Fold 1	85.15	84.34	83.84	82.38	81.40	83.36	84.67	84.39	83.69
Fold 2	84.86	85.59	85.00	83.11	81.22	85.44	86.17	84.71	85.88
Fold 3	81.02	79.21	79.37	80.57	79.37	78.77	80.72	79.22	77.86
Fold 4	80.60	78.54	78.86	77.74	78.37	80.45	76.15	80.76	75.67
Mean	**82.91**	**81.92**	**81.76**	**80.95**	**80.09**	**82.05**	**81.93**	**82.27**	**80.78**

Table 4. 8 Cross validation accuracies for cross-lingual transfer model with different source languages.

From table 4.6, we observed that many of the Indo-European languages experimented on had very good results which were not far from the HMM model's accuracy and they even performed better than the baseline model with all unknowns tagged as unknowns (that is, X). In fact, using English for transition probabilities had a better accuracy than the best baseline model. This is due to a number of possible reasons. One of them may be that using them for training comes with having more training data, and more data is always a good thing in machine learning. The English language has the best accuracy with 82.91% which implies to some extent of it having some similar structures to the Yoruba language, especially in the aspect of word order.

Yoruba, like English language is a S-V-O language (Subject-Verb-Object). This may also be a reason for English having the highest accuracy. Portuguese comes in a close second and the Danish language rounded up the top 3. The Naija pidgin language, which is a local language spoken by many ethnic groups in Nigeria and across Africa was expected to have the best accuracy since it is from the same family of languages as Yoruba. However, it was only 4th best with 81.93% accuracy. This probably proves that two languages coming from the same language family does not mean they are more corelated to each other than another language from a different family of languages.

Like earlier, we explored the confusion matrices together with the precisions, recalls and f1 score in analyzing the errors in the model performance. For this, we used the model with the highest accuracy which is the Italian model for analysis. Figures 4.5 and 4.6 below show the confusion matrices in their numbers and percentages and table 4.9 shows the precision, recall and f1 scores.

Tags/classes	ADJ	ADP	ADV	AUX	CCONJ	DET	NOUN	NUM	PART	PRON	PROPN	PUNCT	SCONJ	VERB
ADJ	4	0	2	0	0	0	2	0	0	1	0	0	0	0
ADP	0	49	0	1	0	2	0	0	0	0	0	0	0	1
ADV	1	1	28	2	0	2	3	0	0	0	3	0	2	3
AUX	0	1	3	48	0	0	0	0	0	0	1	0	0	4
CCONJ	0	0	1	0	35	0	0	0	0	0	0	0	0	0
DET	0	0	3	0	1	14	0	0	0	0	0	0	0	0
NOUN	1	1	2	0	0	8	39	0	0	0	2	0	0	2
NUM	0	0	0	0	0	0	2	2	0	0	0	0	0	0
PART	0	0	2	1	0	0	0	0	1	0	0	0	0	4
PRON	0	1	2	1	0	2	3	0	0	172	0	0	0	7
PROPN	0	0	0	1	0	0	3	0	0	0	42	0	0	0
PUNCT	2	0	0	0	0	0	1	0	0	0	0	169	0	0
SCONJ	0	0	3	8	0	0	0	0	0	0	0	0	55	2
VERB	1	5	7	8	0	7	8	0	0	5	9	0	0	119

Figure 4. 5 Confusion Matrix (a) for the Cross-lingual Tagger

Tags/classes	ADP	ADV	AUX	CCONJ	NOUN	PRON	PROPN	PUNCT	SCONJ	VERB
ADP	5.3	0	0.1	0	0	0	0	0	0	0.1
ADV	0.1	2.7	0.2	0	0.3	0.3	0	0	0.2	3.0
AUX	0.1	0.3	5.2	0	0	0	0.1	0	0	0.4
CCONJ	0	0.1	0	3.8	0	0	0	0	0	0
NOUN	0.1	0.2	0	0	4.2	0	0.2	0	0	0.2
PRON	0.1	0.2	0.1	0	0.3	18.5	0	0	0	0.8
PROPN	0	0	0.1	0	0.3	0	4.5	0	0	0
PUNCT	0	0	0	0	0.1	0	0	18.2	0	0
SCONJ	0	0.3	0.9	0	0	0	0	0	5.9	0.2
VERB	0.5	0.8	0.9	0	0.9	0.5	1.0	0	0	12.8

Figure 4. 6 Confusion Matrix (b) for the Cross-lingual Tagger showing errors (%)

Tags	Precision	Recall	F1 score
ADJ	0.44	0.44	0.44
ADP	0.84	0.92	0.88
ADV	0.53	0.62	0.57
AUX	0.69	0.84	0.76
CCONJ	0.97	0.97	0.97
DET	0.40	0.78	0.53
NOUN	0.64	0.71	0.67
NUM	1.00	0.50	0.67
PART	1.00	0.12	0.22
PRON	0.97	0.91	0.94
PROPN	0.74	0.91	0.82
PUNCT	1.00	0.98	0.99
SCONJ	0.96	0.81	0.88
VERB	0.84	0.70	0.77

Table 4. 9 Precision, Recall and F1 score for the tags with Cross-lingual model

As evident from the classification metrics above, the cross-lingual model performed very well and close enough to the supervised HMM model on some classes such as PUNCT and CCONJ but performed poorly on some other, such as PART, where it had an 0.22 f1 score even though it had a 100% precision but was very poor in recall. In contrast, the supervised HMM model had a 100% f1 score on this same class. The reason for this can be seen from the confusion matrix (a) where out of 8 instances of this class, the model correctly classifies only one of them. In all the models we used so far, VERBs and ADVs consistently appear to be the most misclassified parts of speech with respect to the number of different parts of speech they are consistently confused with. This, of course, does not come much as a surprise as both classes are very similar even in the English language and easily confused for each other. The words responsible for this in the Yoruba tagging are called function words (Oluokun, 2018) as they are highly ambiguous because they can act as parts of speech within a sentence. For example, the word “ni” can act as a verb, Adposition and an auxiliary within a sentence. So, more rules need to be introduced for this kind of words.

To show that indeed training transition probabilities in resource rich languages is much better and has been beneficial, we explored the tagging accuracy for a model with a naïve transition probability distribution which is untrained at all. That is, every POS has a probability of following another POS that is proportional to the relative frequency of all POSs. The result for this is shown in table 4.11 below:

Folds	**Accuracy (%) (Lowercase)**	**Accuracy (%) (Sentence case)**
Fold 1	74.69	75.90
Fold 2	72.81	72.81
Fold 3	79.62	83.99
Fold 4	83.69	84.02
Mean	**77.70**	**79.18**

Table 4. 10 Cross-validation Accuracies for a model with naïve transition probabilities

This is not a surprising result and was expected, this is because the tagger has uniform transition probability distributions for all POSs, that is, untrained in any language, we expected that it should have lower accuracies, and we did get lower accuracies. Also, the lowercase and sentence case differences show here also like in other models. This shows that transferring transition probabilities from resource rich languages in this case has been beneficial to the model, as even the language with the lowest accuracy (Swedish) performs better than the naïve model. In all, we have achieved some decent taggers with good tagging accuracies, however, this is still much far from state of the art tagging accuracies for resource rich languages such as English which has a baseline of 91% for the Brown corpus and over 97% on the Penn Treebank with Neural Networks. However, given the challenges with low resource, our models have done considerably well. The accuracies for different models is summarized in the table below:

Models	Baseline	Supervised HMM	Naïve Transition Model	Cross-lingual Model
Accuracy (%)	82.78	84.64	79.18	82.91

Table 4. 11 Summary of models

4.4 Unsupervised Training

The essence of this training is to find a model that maximizes the probability of observations given our HMM parameters. With that, we can apply the HMM on larger Yoruba corpora which is unannotated and trust the tagging results to a good extent. This is generally an intractable problem as it involves hill climbing and it requires lots of data for effectives which we do not have for training. As has been discussed in chapter 3, its convergence to a local maximum is dependent on its initial conditions via a supervised training. We trained the HMM parameter as we did in earlier models, then using the Baum-Welch algorithm from chapter 3, retrained the parameters to achieve new and improved parameters.

We evaluated the model using our small tagged corpus by testing on our test set using Viterbi algorithm for decoding just like in supervised learning. With this, we achieved a highest accuracy of 75.26% on the training set which is much lesser than we should have. This is given we set a uniform probability for all our transition probabilities as part of the initial conditions. In a case where we used the actual transition probabilities of states, we achieved a much lower accuracy of 26.07%. Again, this shows the importance of getting the initial conditions right and is time consuming. In our E-M tree climbing using multiple iterations, we had to reduce the length of sentences being considered for evaluation. This is because very long sentences push the probability of an observation towards zero which causes our maximization process of the E-M algorithm to break due to division by zero.

Implementing smoothing on probabilities after they had been trained proved abortive as the common smoothing algorithms compute using the counts/frequency of words and not their probabilities. For this reason, we tested on our training set as testing on different possible test sets gives bad results due to multiple unknown words present.

5. CONCLUSION AND FUTURE WORK

5.1 Conclusion

We have presented some of the approaches to part of speech tagging in Natural Language Processing for Low-Resource Languages. We focused on the Yoruba language as a case study analyzed how the different approaches to tagging performed on it. Our highest accuracy was with the supervised Hidden Markov Model without a richer feature of lowercased words, as it ended up being a poorer feature, was 84.64%. We also achieved decent accuracies with other models such as the baseline with 82.78% and the cross-lingual model with 82.91% which was achieved with the English language as a source language for training transition probabilities. This was the highest among all the languages we experimented with.

We believe that this can be further improved with having more annotated data of the Yoruba language and also with algorithms like Neural Networks which has achieved state-of-art results in some resource-rich languages. Also, since we experimented with only languages in the UD treebank with 15 tags which corresponds to our number of tags for the Yoruba language, we cannot authoritatively conclude that the Italian language is the best for the cross-lingual tagger as there maybe another language with 17 tags but would have performed better for training. Finally, our analysis was based on this small annotated corpus of Yoruba, as expected, for larger dataset, a slightly different result may be obtained.

5.2 Future work

We hope to further explore part-of-speech tagging in future work through POS projections with word alignment and unsupervised training for the Yoruba language. Following success with this, we hope to implement POS tagging for a now larger annotated corpus of Yoruba using Neural networks and Bi-LSTMS which have achieved state-of-art results in resource rich languages. We also intend to further this processing to other aspects of Natural Language Processing such as bootstrapping a parser for Yoruba, semantics and machine translation. Finally, we plan to work on other low-resource languages in the nearest future.

6. BIBLIOGRAPHY

Adeniyi, H. R. (2007). A comparative study of reduplication in Edo and Yorùbá. *MorphOn:e-journal of morphology*, pages 1–23.

Adewole, S. M. (1995). *Yoruba Word Formation Processes*. PhD thesis, University of California at Los Angeles.

Awoyale, Y. (1981). Nominal compound formation in yoruba ideophones. *Journal of African Languages and Linguistics*, 3:139–157.

Bamgbose, A. (1966). *A Grammar of Yoruba*. Cambridge University Press, UK.

Bamgbose, A. (2010). *Fonoloji ati Girama Yoruba*. University Press PLC, Ibadan, NIgeria.

Baum, L. E. (1972). An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes. In Shisha, O. (Ed.), *Inequalities III: Proceedings of the 3rd Symposium on Inequalities*, University of California, Los Angeles, pp. 1–8. Academic Press.

Bohnet, B., McDonald, R., Simões, G., Andor, D., Pitler, E., & Maynez, J. (2018). Morphosyntactic Tagging with a Meta-BiLSTM Model over Context Sensitive Token Encodings. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2642–2652

Buys, J., & Botha, J. A. (2016). Cross-Lingual Morphological Tagging for Low-Resource Languages. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1954–1964

Charniack, E., Hendrickson, C., Jackobson, N., Perkowitz, M. (1993). Equations for part-of-speech tagging. In *AAAI-93,* Washington, D.C., pp 784-789. AAAI Press.

Chen, S. F. & Goodman, J. (1998). An empirical study of smoothing techniques for language modeling., Computer Science Group, Harvard University.

Das, D., & Petrov, S. (2011). Unsupervised Part-of-Speech Tagging with Bilingual Graph-Based Projections. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 600–609. Retrieved from https://www.aclweb.org/anthology/P11-1061

Dempster, A. P., Laird, N. M., & *Tech. rep. TR- 10-98*Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society, 39(1), 1–21.

de Marneffe, M.-C., Mac Cartney, B., & Manning, C. D. (2006). Generating typed dependency parses from phrase structure parses. In *LREC.*

de Marneffe, M.-C. & Manning, C. D. (2008). The Stanford typed dependencies representation. *In Workshop on Cross-framework and Cross-domain Parser Evaluation.*

de Marneffe, M.-C., Dozat, T., Silveira, N., Haverinen, K.,Ginter, F., Nivre, J., & Manning, C. D. (2014). Universal Stanford dependencies: A cross-linguistic typology. In *LREC*, pages 4585–4592.

Duong, L., Cook, P., Bird, S., & Pecina, P. (2013). Increasing the Quality and Quantity of Source Language Data for Unsupervised Cross-Lingual POS Tagging. *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, 1243–1249. Retrieved from https://www.aclweb.org/anthology/I13-1177

Fabunmi, F. A. & Salawu, A. S. (2005). Is Yorùbá an endangered language? *Nordic Journal of African Studies*, 14(3):391–408.

Folarin, A.Y. (1989). *Lexical Phonology of Yoruba Nouns and Verbs*. University of Kansas. URL https://books.google.co.uk/books?id=y2HNngEACAAJ.

Francis, W.N (1979) A tagged corpus -problems and prospects. In Greenbaum, S., Leech, G., and Svartvik, J. (Eds) *Studies in English linguistics for Randolph Quirk*, pp. 192-209

Francis, W. N & Kucera, H. (1982) *Frequency Analysis of English Usage.* Houghton Mifflin. Boston

Gale, W. A., Church, K. W., & Yarowsky, D. (1992). Estimating upper and lower bounds on the performance of wordsense disambiguation programs. In *ACL-92*, Newark, DE, pp.

249–256.

Garrette, D., Mielens, J., & Baldridge, J. (2013). Real-World Semi-Supervised Learning of POS-Taggers for Low-Resource Languages. *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 583–592.

Hwa, R., Resnik, P., Weinberg, A., Cabezas, C., & Kolak, O. (2005). Bootstrapping Parsers via Syntactic Projection Across Parallel Texts. *Nat. Lang. Eng.*, *11*(3), 311–325. https://doi.org/10.1017/S1351324905003840

Jelinek, F. & Mercer, R. L. (1980). Interpolated estimation of Markov source parameters from sparse data. In Gelsema, E. S. and Kanal, L. N. (Eds.), *Proceedings, Workshop on Pattern Recognition in Practice*, pp. 381–397. North Holland, Amsterdam.

Johnson, M. (2007). Why doesn't EM find good HMM POS-taggers. *In Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pp. 296–305, Prague.

Jurafsky, D., & Martin, J. H. (2000). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition* (1st ed.). Upper Saddle River, NJ, USA: Prentice Hall PTR.

Klein, S. & Simmons, R. F. (1963). A computational approach to grammatical coding of English words. *Journal of the Association for Computing Machinery*, *10*(3), 334–347.

LEWIS, P. (2009).Ethnologue: Languages of the World. SIL International.

Ling, W., Dyer, C., Black, A. W., Trancoso, I., Fermandez, R., Amir, S., … Luís, T. (2015). Finding Function in Form: Compositional Character Models for Open Vocabulary Word Representation. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 1520–1530. https://doi.org/10.18653/v1/D15-1176

Li, S., Graca , J., & Taskar, B. (2012). Wiki-ly supervised part-of-speech tagging. *In Proceedings of EMNLP*, Jeju Island, Korea.

Marcus,M. P., Santorini, B., & Marcinkiewicz, M. A. (1993). Building a large annotated corpus of English: The Penn treebank. *Computational Linguistics*, *19*(2), 313–330.

Ma, X., & Xia, F. (2014). Unsupervised Dependency Parsing with Transferring Distribution via Parallel Guidance and Entropy Regularization. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1337–1348. https://doi.org/10.3115/v1/P14-1126

Mikolov, T., Kombrink, S., Karafiát, M., & Burget, L. (2011). Recurrent Neural Network Based Language Modeling in Meeting Recognition. *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, 2877–2880.

Nivre, Joakim; Abrams, Mitchell; Agić, Željko; et al., (2018). Universal Dependencies 2.3, LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University, http://hdl.handle.net/11234/1-2895.

Oluokun, A. (2018) Creation of a Dependency Treebank for Yoruba using Parallel Data. *Institute of Formal and Applied Linguistics.* Prague.

Pan, S. J., Yang, Q., Fan, W., & D, S. J. P. (ph. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*.

Petrov, S., Das, D., & McDonald, R. (2012). A Universal Part-of-Speech Tagset. *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)*, 2089–2096. Retrieved from http://www.lrec-conf.org/proceedings/lrec2012/pdf/274_Paper.pdf

Plank, B., & Agić, Ž. (2018). Distant Supervision from Disparate Sources for Low-Resource Part-of-Speech Tagging. *ArXiv:1808.09733 [Cs]*. Retrieved from http://arxiv.org/abs/1808.09733

Sukhareva, M., Fuscagni, F., Daxenberger, J., Görke, S., Prechel, D., & Gurevych, I. (2017). Distantly Supervised POS Tagging of Low-Resource Languages under Extreme Data Sparsity: The Case of Hittite. *Proceedings of the Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, 95–104.

Tackstrom, O., Das, D., Petrov S., McDonald, R., & Nivre, J., (2013). Token and Type Constraints for Cross-lingual Part-of-speech Tagging. *Transactions of the Association for Computational Linguistics*, 1:1–12

UCLA. UCLA Language Materials Project: Yoruba, n.d. Retrieved June 3, 2019, from http://www.lmp.ucla.edu/Profile.aspx?menu=004&LangID=22.

Welch, L. R. (2003). Hidden Markov Models and the Baum-Welch Algorithm. *IEEE Information Theory Society Newsletter,* 53(4):1–24, December

Witten, I. H. & Bell, T. C. (1991). The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. *IEEE Transactions on Information Theory*, *37*(4), 1085–1094.

Yarowsky, D. & Ngai, G. (2001). Inducing multilingual POS taggers and NP bracketers via robust projection across aligned corpora. *In Proc. of NAACL.*

Zeman, D. (2008) Reusable tagset conversion using tagset drivers. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, and Daniel Tapias, editors, *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco. European Language Resources Association (ELRA). ISBN 2-9517408-4-0. http://www.lrecconf.s org/proceedings/lrec2008/.

Zennaki, O., Semmar, N., & Besacier, L. (2015). Unsupervised and Lightly Supervised Part-of-Speech Tagging Using Recurrent Neural Networks. *Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation*, 133–142. Retrieved from https://www.aclweb.org/anthology/Y15-1016