

# Carl Bildt Tweets: A comparison of regular and constrained Markov chain for text generation

Group Ain't intelligent

Viktor Björkholm	Jesper Bränn	Daniel Duberg	Jakob Tideström
92-11-17	92-09-30	93-01-13	90-10-04
viktorbj@kth.se	jbrann@kth.se	dduberg@kth.se	jakobti@kth.se



## Abstract

Skriv sist, nr vi vet vad vi har skrivit om (y)

# 1 Introduction (1–2 pages)

**' What is the problem and why is it important?**

**What will the reader get out of it?**

**Present specific results**

**Don't go into detail**

This paper aims to develop an understanding on refining natural language text generation. Natural Language Generation (NLG) is an area of research within the field of Artificial Intelligence. The aim is to generate text that is semantically correct in order to make communication with computer systems more natural and understandable for users.

Within this paper we show the difference in quality of two different approaches to text generation. One of these approaches is using Markov chains or more commonly known as n-grams. These n-grams take n words in sequence and uses a corpus of text to guess what the most probable next word is. Using a larger n means that more text is copied straight from the corpus, however this also means that there is a higher likelihood that the text being generated is meaningful. We will contrast this method with using constrained Markov chains. The main constraint of the Markov chains is that two words following each other will have the same sequence of part-of-speech as the corpus. Part-of-speech is a concept within NLG that divides a text into the different linguistical categories of the words within it.

We aim to show that using this constraint upon the Markov chain, sentences will have a greater diversity but still be as semantically correct as just using a n-gram. Since the problem with larger n:s within n-grams is that text is copied straight from the corpus, the constraint will help us create semantically correct sentences without taking word sequences straight from the corpus.

To be able to show differences in these two approaches we generate Twitter messages, so called tweets. We build upon the work by Barbieri et al., 2012 to implement our own constraints.

## 1.1 Contribution

forklara för en datalog typ varför det vi gjort är vettigt.

We have implemented a unigram of part-of-speech on to a bigram in order to observe the difference in the result. As mentioned above a problem with n-grams with to high n:s is that they will simply copy parts of the corpus if said corpus does not contain a large variation of similar sentences, so that a given start of n words does not automatically lead to one sentence finish. i.e. a larger corpus is needed for larger n:s. To be able to keep a smaller and diverse corpus we applied the unigram ontop of the n-gram to allow the program to select words of a common word type order but perhaps not words that have occurred after each other naturally in the corpus.

Our main contribution to the field is that we have tried putting this unigram constraint upon the regular Markov chain and doing it for short message generation. The research area we base this paper on focus on generating text that fits a theme, or rhythm whereas we generalize the concept.

A lot new research involves the constraining of Markov chains to produce text that fits different molds than just text generated from a corpus. It is important to be able to generate text that is

## 1.2 Outline

We bring up the relation of our work to some other work in the field Barbieri et al. [2012] in section 2 and explain how that research has affected ours. In section 3 we then go through the details on how the algorithm works, we also give examples to explain in detail what the difference between the two methods we are comparing. This is complemented with details regarding our specific implementation of the algorithm in section 3.1. The data from running the algorithm is explained, reviewed and evaluated in section 4.

## 2 Related work (1–3 pages)

**It is here we show that we know the field well and present the reports that we have read**

Our work on the constrained Markov models are built upon Barbieri et al. [2012] work, where the authors generated lyrics from different artists using Markov models with constraints. These were to be generated in a specific style and with a rhythm, to simulate real lyrics from a large span of artists. They start in a standard fashion by building up a corpus of content that they want to imitate. In their case they built up a corpus consisting of the collected work of each artist. So when generating lyrics for bob dylan as an example, the corpus consisted of 7600 unique words, a total of 96 089 words and 12 408 verses. They proceeded with applying constraints on a bigram to meet their demands on the generated text to have a certain style and to rhyme. Some of the constraints they apply on their bigram is that some words need to have the same meter as in the original song, but also that the text generated needs to fit a part-of-speech template and a rhythmic template that comes from the song the newly generated lyrics is to be based on. The templates are not generated by their algorithm, but rather handcrafted to get a proper result. Our take from this paper was to be able to apply constraints upon a bigram, or rather a Markov chain in general. Similarly we utilize a part-of-speech template of sorts, but we diverge a bit from this paper.

**Saker vi behver svara p i detta stycke:**

Vad vet vi om hur dem anvnde constraints i sitt arbete? Vi verkar ju inte veta ngot om det eftersom vi bara gissar oss fram :D

Har vi ngot mer att basera vratt arbete p? En vetenskaplig text r ju bra, men om vi lst fler s r ju det rtt bra.

### 3 My method (1–4 pages)

Our method that forms the basis of this paper is generating Twitter messages with the use of both Markov chains and constrained Markov chains and to compare the two methods with each other. The theory is that constraining an Markov chain will yield better and more diverse text being generated. In section 4 where we explain our experiments and data we try out different orders of the Markov chain, for the sake of explanation we assume a first order Markov chain is used to explain how the method works. In a limited corpus such as this one:

“Rolf has a dog.”	Noun → verb → singular quantifier → noun
“Rolf owns a dog.”	Noun → verb → singular quantifier → noun
“Rolf can not walk his dog.”	Noun → modal → adverb → verb → pronoun → noun

This corpus generates a Markov chain that looks like figure 1a, with the edges being the probabilities for the specific transitions. In the figure the specific part-of-speech is included in the states beneath the words from the corpus. If we then add constraints from a transition matrix, built up from POS-analysis of the same corpus we are given the new chain that is seen in figure 1b.

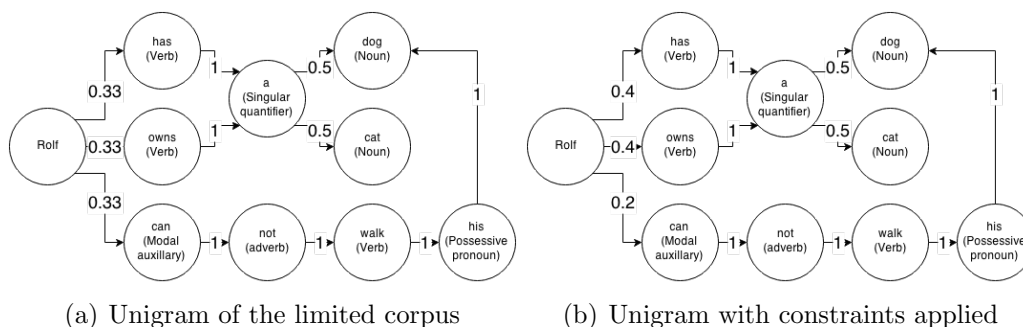


Figure 1: Unigrams

We can see in figure 1b that since both ”has“ and ”owns“ are verbs they are more probable to occur than ”can“, this is reflected in the new edges. This happens because the part-of-speech ”verb“ is more likely to follow NNP according to our transition matrix, and thus ”has“ and ”owns“, who are both verbs become even more likely to follow NNP (Rolf). This method can then be further applied to a bigram, a trigram or any n-grams that follows. Our method however will only have an unigram for the transition matrix, even if

the Markov chain of words from the corpus is longer, the transitions are only observed with one previous state in consideration.

**Corpus:**

In order to generate tweets, we first discussed the problem regarding the semantical corectness of a generalized tweet. The average level is according to our own experience far from sematically correct, which isn't a problem in understadability for an experienced Twitter user, but is a problem for our POS who would not recognize the words. Even if it would give it an "unknown"-tag, we would not be able to predict any kind of results.

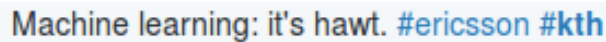
The image shows a single line of text from a tweet. The text is "Machine learning: it's hawt. #ericsson #kth". The words "Machine learning:" are in a dark blue font, "it's hawt." is in a lighter blue font, and the hashtags "#ericsson" and "#kth" are in a darker blue font. The entire text is set against a light blue background.

Figure 2: An example tweet

To solve this problem approximatily, we decided to generate tweets for a specific user who mostly uses correct grammar and semantics when tweeting and tweets in english. Our option fell on Carl Bildt, former foreight minister of Sweden, because of his active use of Twitter, that he tweets in english and that most of his tweets are in gramatically correct english.

The Markov chain is built up with the use of n-grams of the second order, so called bigrams. This means in a limited corpus like for example “Rolf has a dog.” “Rolf owns a dog.” “Rolf can not walk his dog.” Would generate a

	Rolf	→	has
	has	→	a
	a	→	dog
	Rolf	→	owns
	owns	→	a
bigram that would look like this:	a	→	cat
	Rolf	→	can
	can	→	not
	not	→	walk
	walk	→	his
	his	→	dog

We see here that the probability of nice coming after the words ”a“ and ”very“ is 100%, since that is the only combination that happens. However, the probability of ”town“ coming after ”very“ and ”nice“ is only 50%.

Our method consisted of building a transition matrix as a unigram of parts-of-speech (POS) together with a bigram with actual words from our corpus of text. From the transition matrix we generated a constrained transition matrix based on the amount of number of words we wanted the tweet to contain (**and other criteria**). The constrained matrix was generated using the method described in the work by Barbieri et al. [2012]. We generalized the method and made it work for our transition matrix even though it consisted only of parts-of-speech.

The different constrains for creating a tweet that we had to consider were that they can not be longer than 140 characters, they have to end with an end-symbol (dot, exclamation mark or question mark) and they should probably have a reasonable minimum length. When iterating through a corpus we are using a POS-tagger to identify the different types of words and coding to be able to build a transition matrix for the sequences for the different word types (the probability for a noun to be followed by a word for an example). The next step is to implement our constraints on the transition matrix. The constraint for the length of the tweet caused us problems since we only know the types of the words and would not be in touch with the length of the specific words that are chosen for our tweet. We decided to approximate this constraint to a limited number of words in our tweet.

#yolo

Bla bla	Bla bla	Bla bla
42	42	42
42	42	42

[illegible]



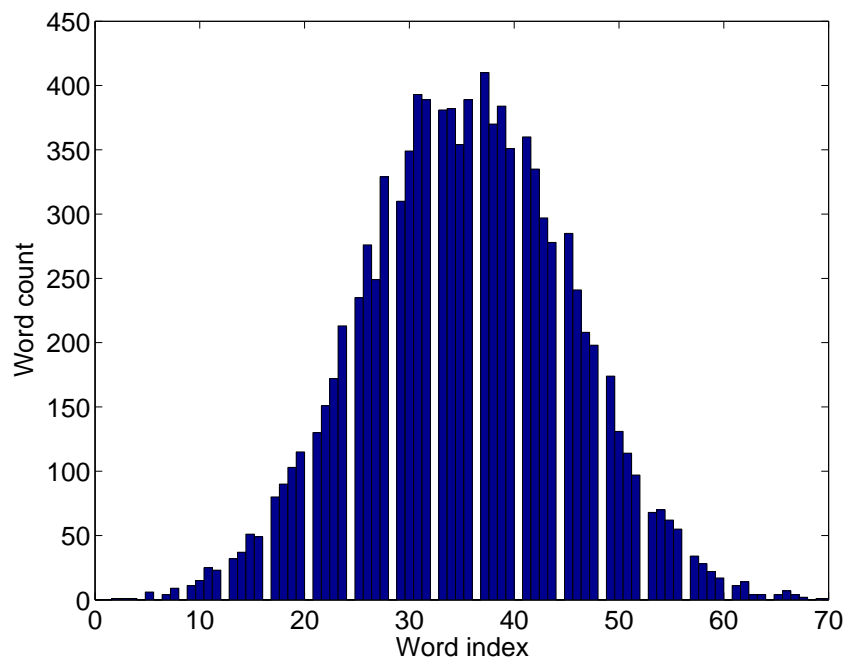


Figure 3: A description that makes browsing the paper easy and clearly describes what is in the picture. Make sure that the text in the figure is large enough to read and that the axes are labelled.

## 5 Summary and Conclusions (0.5–1 page)

Bla bla  
bla  
bla bla bla bla bla bla bla bla bla bla bla bla bla bla

## References

Gabriele Barbieri, Francois Pachet, Pierre Roy, and Mirko Degli Esposti.  
Markov constraints for generating lyrics with style. 242:115–120, 2012.