Search Medium Write

# Leveraging N-grams to Extract Context From Text

A simple intro to a basic but fundamental NLP concept

Aashish Nair · Following
Published in Towards Data Science · 5 min read · Nov 2, 2021

53 1

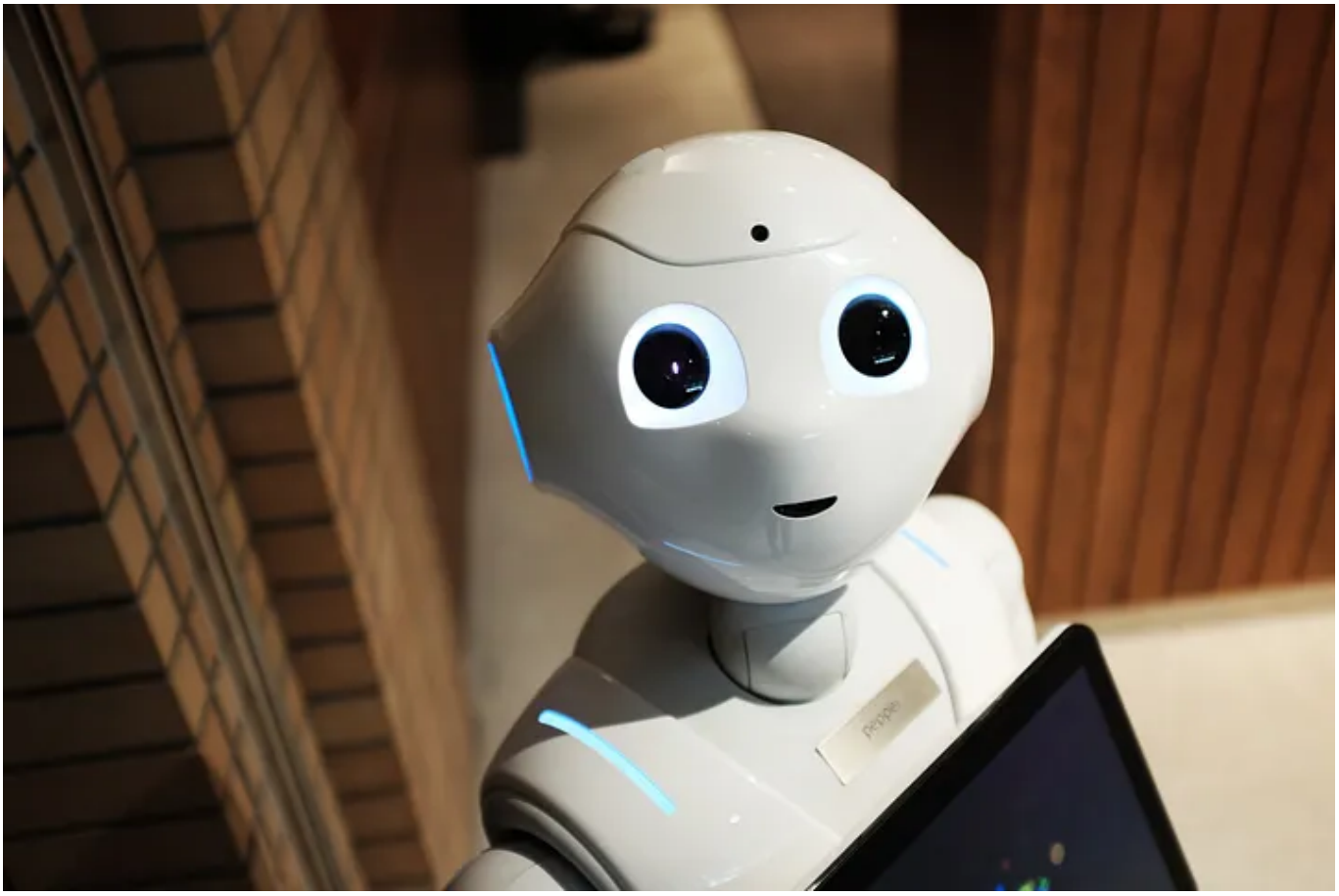


Photo by Alex Knight from Pexels

People will inevitably run into n-grams when learning to deal with textual data. They often play a key role in enabling machines to understand the context of the given text.

As a result, this term is brought up in countless data science projects.

However, instead of treating n-grams as jargon that one can simply gloss over, it is important to take the time to learn about the ins and outs of this concept since it will serve as the foundation for understanding more advanced natural language processing tools and techniques.

So, what are n-grams?

Simply put, n-grams refer to a sequence of N words or characters.

## Example

Let's consider the sentence: "I live in New York".

A unigram model (n=1), stores this text in tokens of 1 word:

["I", "live", "in", "New", "York"]

A bigram model (n=2) stores this text in tokens of 2 words:

["I live", "live in", "in New", "New York"]

In this scenario, the city "New York" would not be recognized as an entity with the unigram since each token only stores one word. On the other hand, the bigram joins the words "New" and "York" and allows the machine to

recognize “New York” as a single entity, thereby extracting the context from the text.

## Applications

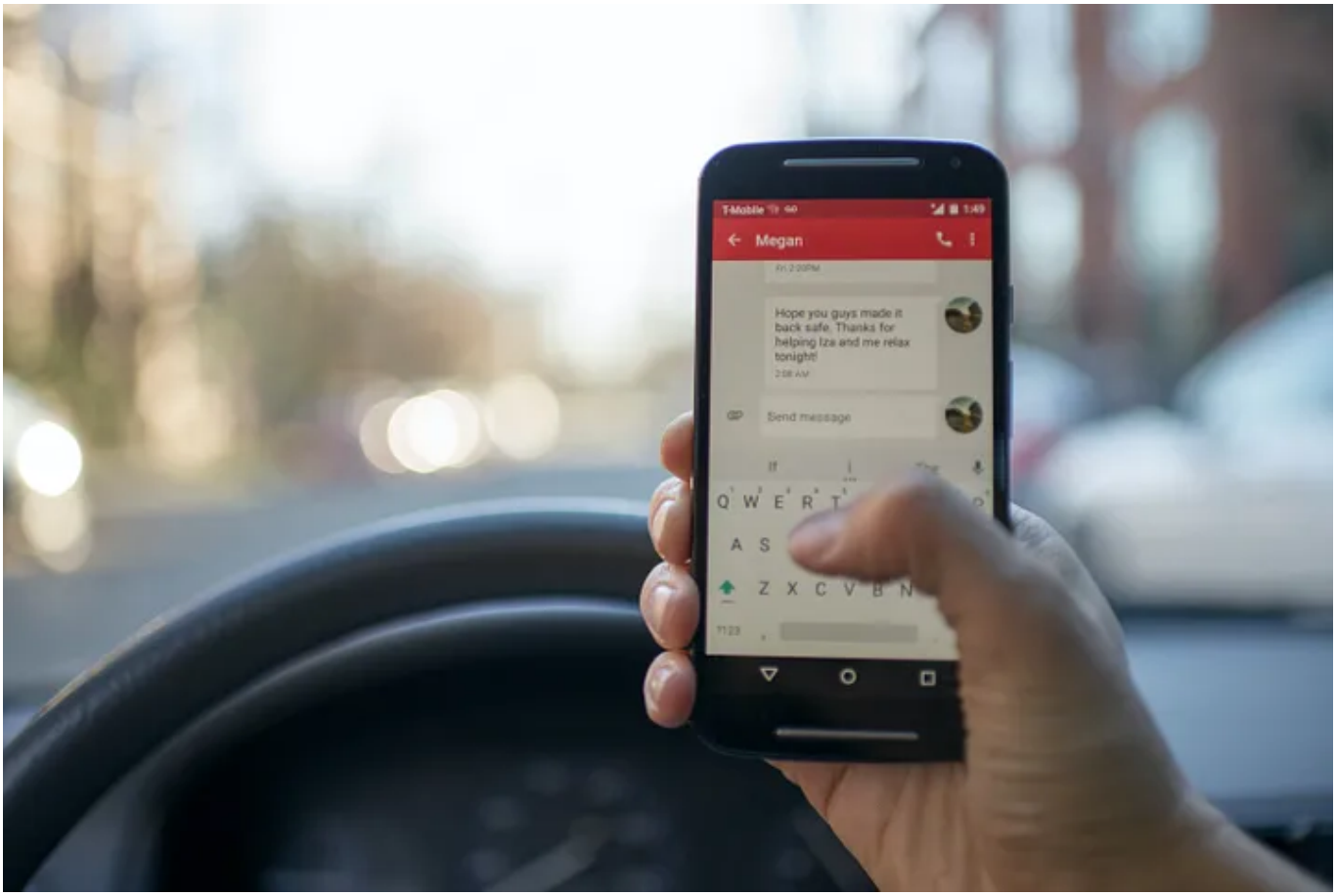



Photo by Roman Pohorecki from Pexels

N-grams are ubiquitous in natural language processing.

Think of the text suggestion features in your messengers or search engines that you have learned to take for granted. Think of the spam detectors or hate speech detectors that make your experience with social media more pleasant. These features all rely on n-grams to achieve the reliability that they have become known for.

With that being said, there is no specific n-gram model that trumps all the others. The best way to leverage n-grams in your NLP models can only be determined through experimentation.

## Case Study

Let's do a case study to demonstrate the impact n-grams have on model performance.

Here, we will conduct a simple sentiment analysis using a dataset containing reviews of video games (accessible here) along with their corresponding sentiment score.

Loading Dataset

| | 2 | 3 |
|---|---|---|
| 0 | Positive | im getting on borderlands and i will murder yo... |
| 1 | Positive | I am coming to the borders and I will kill you... |
| 2 | Positive | im getting on borderlands and i will kill you ... |
| 3 | Positive | im coming on borderlands and i will murder you... |
| 4 | Positive | im getting on borderlands 2 and i will murder ... |

Code Output (Image By Author)

The data is first cleaned prior to any predictive modeling.

Data Preprocessing

For this case study, the text will be converted to a bag of words with the CountVectorizer object in the sklearn module before being used to train a machine learning classifier.

Bag Of Words With Unigrams

Note: The “ngram_range” parameter refers to the range of n-grams from the text that will be included in the bag of words. An n-gram range of (1,1) means that the bag of words will only include unigrams.

Let’s see how a Naive Bayes model predicts the sentiment of the reviews with an n-gram range of (1,1).

Training a Model With Unigrams

```
F-1 score : 0.752
```

Code Output (Image By Author)

The Naive Bayes classifier registers an f1-score of 0.752.

In general, the bag of words model is a very simple approach to word vectorization and has certain limitations. Mainly, the words are not ordered as they are collected, so a lot of the context from the text is lost.

Using n-grams, in this case, can help address this issue by establishing some order to preserve context.

Let's see how the model performs after adding bigrams (n=2) to the input features. This can be achieved by changing the "ngrams_range" parameter to (1,2).

Training a Model With Unigrams and Bigrams

```
F-1 score : 0.882
```

Code Output (Image By Author)

The model registers a greater f-1 score after the inclusion of bigrams. This can be attributed to the greater context the machine gets when it inputs 2-word sequences instead of just individual words.

That being said, when it comes to n-grams, more is not necessarily better. In some cases, having too many features will result in a less optimal model.

We can prove this by showing how accurately the model predicts the sentiment of the given text with different n-gram ranges.

Here, 10 models with different n-gram ranges are built with the training set and evaluated with the testing set.

Training a Model With 10 N-gram Ranges

```
F-1 score of model with n-gram range of (1, 1): 0.752
F-1 score of model with n-gram range of (1, 2): 0.882
F-1 score of model with n-gram range of (1, 3): 0.8994
F-1 score of model with n-gram range of (1, 4): 0.9017
F-1 score of model with n-gram range of (1, 5): 0.9019
F-1 score of model with n-gram range of (1, 6): 0.9011
F-1 score of model with n-gram range of (1, 7): 0.9006
F-1 score of model with n-gram range of (1, 8): 0.9005
F-1 score of model with n-gram range of (1, 9): 0.9001
F-1 score of model with n-gram range of (1, 10): 0.8998
```

Code Output (Image By Author)

Based on the results, the model performs at its best with the n-gram range of (1,5). This means that training the model with n-grams ranging from unigrams to 5-grams help achieve optimal results, but larger n-grams only result in more sparse input features, which hampers model performance.

## Conclusion


Photo by Prateek Katyal from Pexels

N-grams are a simple but important concept in natural language processing. Given the variety of applications that require extracting insights from text, n-grams will no doubt play a big role in many machine learning projects.

That is why it is essential to gain a strong understanding of n-grams and its impact on model performance. Harnessing this tool will only improve the capability of your models.

I wish you the best of luck in your machine learning endeavors!

Data Science | Naturallanguageprocessing | Machine Learning | NLP | Ngrams

## More from the list: "NLP"

Curated by Himanshu Birla

Jon Gi... in Towards Data ...

**Characteristics of Word Embeddings**

· 11 min read · Sep 4, 2021

Jon Gi... in Towards Data ...

**The Word2vec Hyperparameters**

· 6 min read · Sep 3, 2021

Jon Gi... in

**The Word2ve**

· 15 min rea

View list


## Written by Aashish Nair

Following

1.1K Followers · Writer for Towards Data Science

Data Scientist aspiring to teach and learn through writing. Reach out to me on LinkedIn: www.linkedin.com/in/aashish-nair.

## More from Aashish Nair and Towards Data Science

Aashish Nair in Towards Data Science

## Targeting Multicollinearity With Python

Learn how Python's features help deal with this 8-syllable enigma with ease

6 min read · Dec 6, 2021

138 2

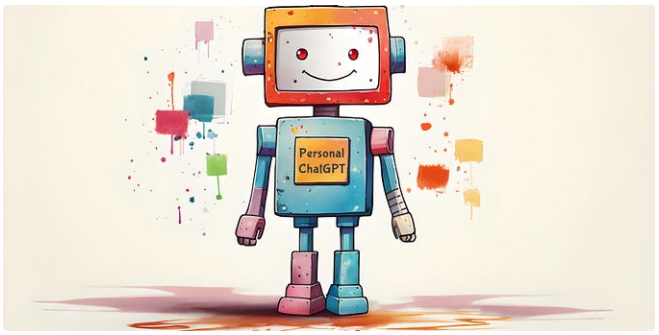

Antonis Makropoulos in Towards Data Science

## How to Build a Multi-GPU System for Deep Learning in 2023

This story provides a guide on how to build a multi-GPU system for deep learning and...

10 min read · Sep 17

549 11



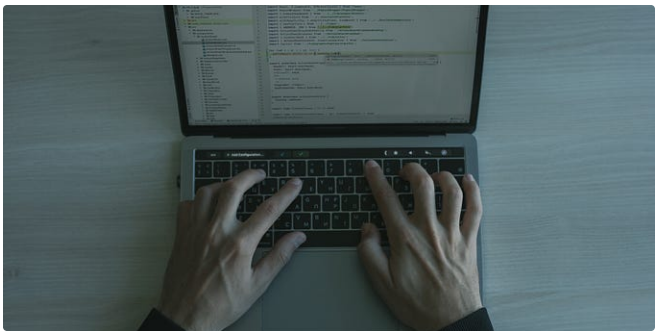

Robert A. Gonsalves in Towards Data Science

## Your Own Personal ChatGPT

How you can fine-tune OpenAI's GPT-3.5 Turbo model to perform new tasks using you...

· 15 min read · Sep 8

595 7



Aashish Nair in Towards Data Science

## How to Improve ChatGPT's Generated Code with Prompt...

A simple strategy for enhancing ChatGPT's performance as your coding assistant

· 7 min read · Jul 21

98 1

See all from Aashish Nair

See all from Towards Data Science

## Recommended from Medium


Rashini Liyanarachchi in Text Mining Basics

### RegEx for Text Mining—I

RegEx or Regular Expressions are used in Text Mining mainly for Basic Patterns or...

7 min read · Jul 25

1

Sanjithkumar

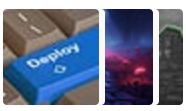
### Text Preprocessing Part—2

Text preprocessing is an integral part of Natural Language Processing as no machine...

5 min read · Sep 7

27 1

## Lists



**Predictive Modeling w/ Python**

20 stories · 452 saves

**Natural Language Processing**

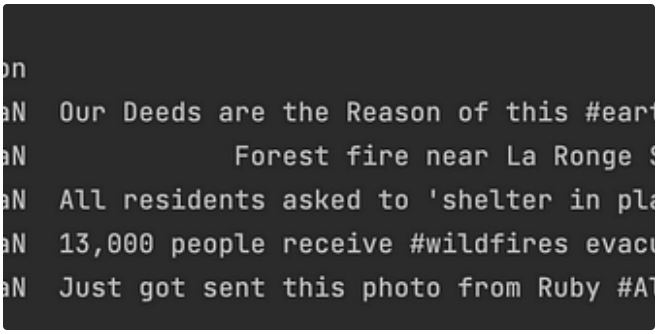
669 stories · 283 saves

**Practical Guides to Machine Learning**

**New_Reading_List**

174 stories · 133 saves

10 stories · 519 saves




Hatice Şeyma Koç

## Fasttext & Doc2Vec for Text Classification

Hello everyone,

5 min read · Jul 7

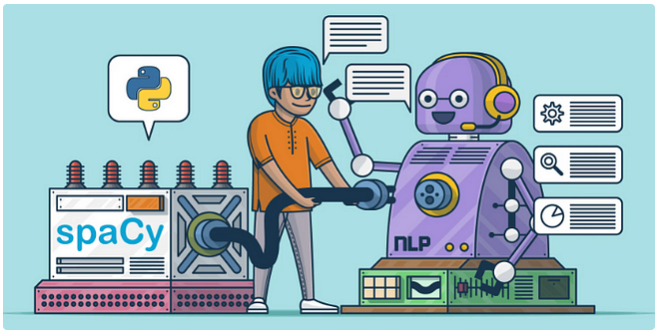
11


Varrel Tantio

## Understanding TF-IDF in Natural Language Processing

Natural Language Processing (NLP) is a rapidly evolving field in the realm of...

3 min read · Sep 22




HasancanÇakıcıoğlu

## Comprehensive Text Preprocessing NLP (Natural Language...

Text preprocessing plays a crucial role in Natural Language Processing (NLP) by...

12 min read · Jul 9

Mustafa Germec, PhD in Python in Plain English

## Text Preprocessing with Natural Language Processing (NLP)

Mustafa Germec, PhD

20 min read · Sep 26


See more recommendations