Member-only story

Attempting to Bring Order to Chaos: Clustering Medium Article Titles with DistilBERT

Tarek · Follow

Published in Level Up Coding · 5 min read · Apr 20

Photo by Mel Poole on Unsplash

In this blog post, we'll introduce you to "DistilBERT," a variant of BERT, and show you how to combine it with Gaussian Mixture Models to cluster medium article titles. We will use real examples from popular publications like "Towards Data Science" and "The Startup," you'll gain a better understanding of how this powerful technique can be used to automatically organize titles.

What is DistilBERT?

DistilBERT is a lightweight version of BERT (Bidirectional Encoder Representations from Transformers), a pre-trained language model that uses a transformer architecture to understand the context of words in a sentence. DistilBERT was created by compressing the original BERT model, resulting in a smaller and faster model that maintains its accuracy and effectiveness.

Although DistilBERT may not match BERT's performance in more complex language tasks, it should suffice for title clustering purposes.

What is Gaussian Mixture Models (GMM)?

Gaussian Mixture Models (GMM) is a clustering algorithm that is used to group data points into clusters based on their similarity. The algorithm is based on the assumption that the data points in a cluster are normally distributed, and the algorithm tries to find the parameters that define these distributions.

Combining DistilBERT and GMM for Title Clustering

By combining DistilBERT and GMM, we can automate the process of categorizing article titles. We first use DistilBERT to extract meaningful representations of each title, which can then be used by GMM to cluster the titles based on their similarity.

To demonstrate this process, we will use a dataset of medium article titles from two publications “Towards Data Science”, “UX Collective” and “The Startup.”

Running DistilBERT to Extract Title Embeddings

```
import torch
import numpy as np
from transformers import AutoTokenizer, AutoModel

model_name = 'distilbert-base-cased'
tokenizer = AutoTokenizer.from_pretrained(model_name)
model = AutoModel.from_pretrained(model_name)

title_embeddings = []
for title in titles:
    encoding = tokenizer.encode_plus(
        title,
        add_special_tokens=True,
        max_length=128,
        pad_to_max_length=True,
        return_attention_mask=True,
        return_tensors='pt'
    )
    with torch.no_grad():
        embedding = model(encoding['input_ids'], encoding['attention_mask'])[0][
        title_embeddings.append(embedding.numpy())
title_embeddings = np.array(title_embeddings).squeeze()
```

In this code, we utilize the DistilBERT model to generate embeddings for a list of medium article titles. It first tokenizes each title using the model’s tokenizer, and generates an embedding for each title using the DistilBERT model. The resulting embeddings can be used for various natural language processing tasks, such as clustering similar article titles which we will do next.

Clustering the Titles

```
from sklearn.mixture import GaussianMixture
clusters = GaussianMixture(n_components=3).fit_predict(title_embeddings)
```

This Python code uses the Gaussian Mixture Model (GMM) algorithm to cluster the embeddings generated by DistilBERT. The code first specifies the number of clusters to generate using the GMM algorithm. It then fits the GMM algorithm to the title embeddings, generating cluster assignments for each title. These assignments are added to a list, which can be used for various natural language processing tasks, such as grouping similar article titles together.

Next, we assign each title to its cluster list using the following code:

```
title_clusters = []
for i in range(len(np.unique(clusters))):
    cluster_titles = []
    for j, title in enumerate(titles):
        if clusters[j] == i:
            cluster_titles.append(title)
    title_clusters.append(cluster_titles)
```

Results

To get a sense of the clustering quality, we can inspect the first 5titles in each cluster:

```
for idx, cluster in enumerate(title_clusters):
    print(f'Cluster {idx}:')
    print('\n'.join(np.random.choice(cluster, 5)))
    print()
```

This will return:

```
Cluster 0: [Seems to be related to Towards Data Science topics]
Visualized Linear Algebra to Get Started with Machine Learning: Part 2
Hacking Causal Inference: Synthetic Control with ML approaches
Performance Estimation Techniques for Machine Learning Models
Claymorphism in user interfaces
How I Address Direct Questions on My Competitors as a 1-Man Consultant

Cluster 1: [Seems to be related to UX Collection topics]
Screen time: the next plastic?
Google wants you to test LaMDA; how UX research can help it outperform
The Vignelli Canon: A design classic from the last of the modernists
Standing at the crossroads of authenticity and career advancement
Product design is going down a weird path, but we can still save it

Cluster 2: [Seems to be related to The Startup topics]
I Tried Substack for a Month, You Won't Believe What Happened
What I'm Doing as the Recession Gets Worse (To Avoid Going Broke)
Your Results Will Only Change When You Do
5 New Side Hustles You Didn't Even Know Existed
4 Easy Tips To Make the Most Out of LinkedIn as a Newbie Writer
```

For a more aesthetically pleasing summarization of each cluster, we can use word clouds:

```
def show_word_cloud(cluster):
    words = [word.lower() for title in cluster for word in title.split() if "'"
    text = " ".join(words)
    wordcloud = WordCloud(width = 800, height = 800,
```

```
                        background_color ='white',
                        stopwords = STOPWORDS,
                        min_font_size = 10).generate(text)

    plt.figure(figsize = (2, 2), facecolor = None)
    plt.imshow(wordcloud)
    plt.axis("off")
    plt.tight_layout(pad = 0)

    plt.show()
```

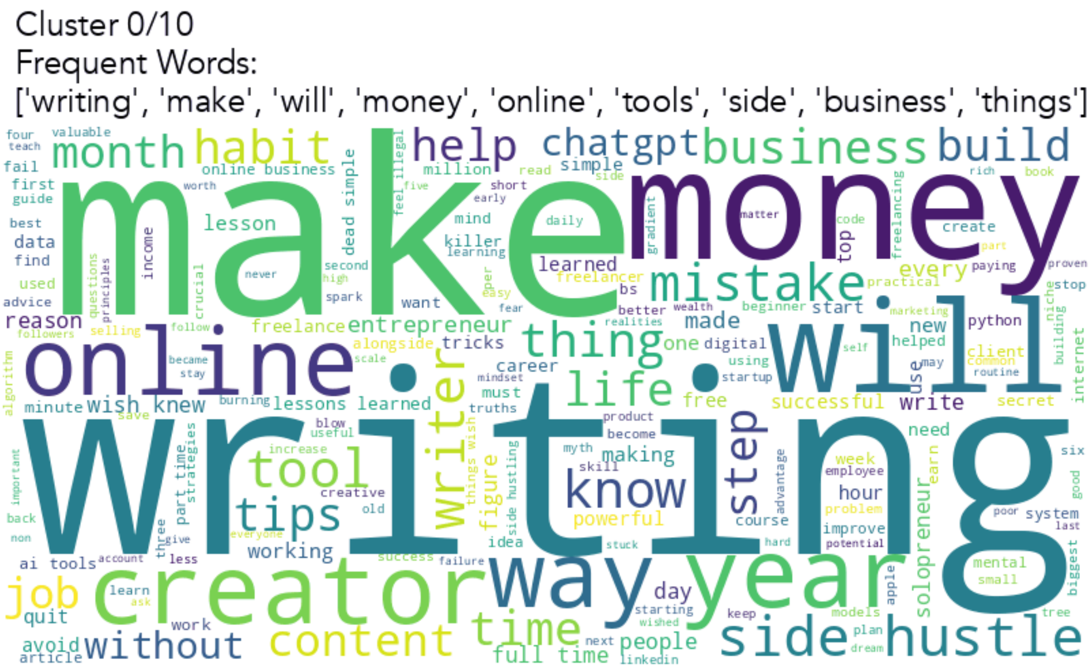
This code takes a cluster as input and plots a word cloud, here you can see all generated word clouds:

WordClouds for all three clusters.

Can you take a guess and pair each cluster with its corresponding publication? ;)

Want More Clusters?

If you would like to see how the quality changes when using more clusters? Check out this GIF that demonstrates clustering with 10 clusters.



WordClouds for 10 clusters with frequest words in each.

Level Up Coding

Thanks for being a part of our community! Before you go:

- 👏 Clap for the story and follow the author 👉
- 📰 View more content in the [Level Up Coding publication]
- 💰 Free coding interview course ⇒ [View Course]
- 🔔 Follow us: [Twitter] | [LinkedIn] | [Newsletter]

🚀👉 **[Join the Level Up talent collective and find an amazing job]**

Machine Learning | Clustering | Bert

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 Tarek

Follow

169 Followers · Writer for Level Up Coding

As a software engineer, I have a passion for exploring various technical topics and blogging about them.

More from Tarek and Level Up Coding

Tarek in The Pythoneers

From Tqdm to Rich: My Quest for Better Progress Bars

Rich's progress bar library, maybe unpopular compared to tqdm yet it is worth checking...

· 2 min read · May 11

45

Victor Timi in Level Up Coding

"Good Commit" vs "Your Commit": How to Write a Perfect Git Commi...

A good commit shows whether a developer is a good collaborator — Peter Hutterer, Linux.

· 8 min read · Sep 5

2.6K 32

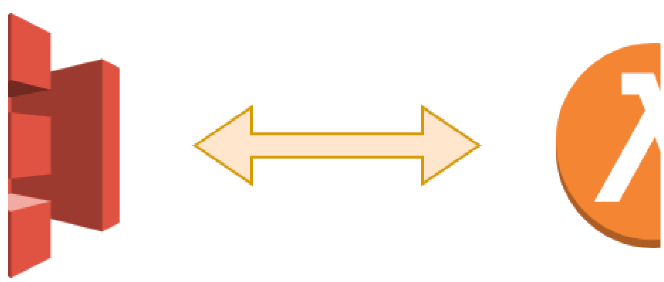
Arslan Ahmad in Level Up Coding

12 Microservices Patterns I Wish I Knew Before the System Design...

Mastering the Art of Scalable and Resilient Systems with Essential Microservices Desig...

13 min read · May 16

3.8K 17



Tarek in The Pythoneers

Parallel Processing on S3: How Python Threads Can Optimize Yo...

Using Threads for Reading and Writing Data on S3: Scenarios, Implementation, and...

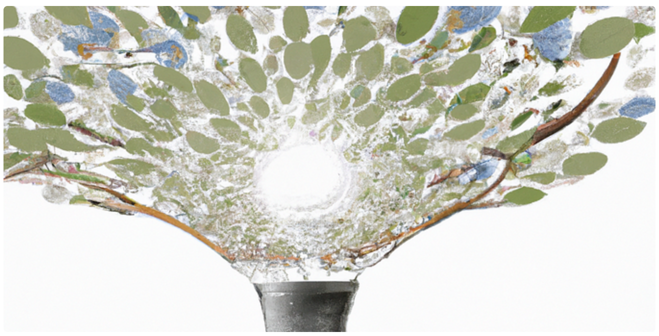
· 4 min read · Apr 15

35 1

See all from Tarek

See all from Level Up Coding

Recommended from Medium



Sirsh Amarteifio

Cluster chatter: HDBSCAN + LLM

HDBSCAN is a density based (hierarchical) clustering algorithm. Clustering algorithms...

5 min read · Jun 13

3

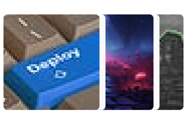
YashwanthReddyGoduguchintha

K-means, kmodes, and k-prototype

K-means, kmodes, and k-prototype are all types of clustering algorithms used in...

9 min read · Apr 10

Lists

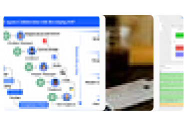


Predictive Modeling w/ Python

20 stories · 452 saves

Practical Guides to Machine Learning

10 stories · 519 saves



Natural Language Processing

669 stories · 283 saves

The New Chatbots: ChatGPT, Bard, and Beyond

13 stories · 133 saves

TechClaw

Cosine similarity between two arrays for word embeddings

Introduction

2 min read · Jul 11

Shruti Dhumne

Mean-Shift Clustering: A Powerful Technique for Data Analysis with…

Introduction

3 min read · Jun 1

Deniz Gunay

Clustering

Clustering

20 min read · Sep 18

Alyx

Semantic Search with FAISS

HuggingFace get_neareast_example and Cosine Similarity Search

9 min read · Jul 15

1

See more recommendations