

Fine-tuning a model

WORKING WITH HUGGING FACE



Jacob H. Marquez
Lead Data Engineer

What is fine-tuning?

- Fine-tuning: adjusting a pre-trained model for a specific task or dataset
- Pre-trained model: algorithm developed on extensive data to perform a task



Why fine-tune?

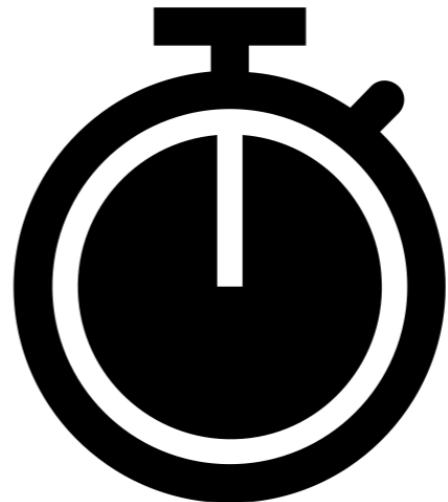


Learn new
task or
domain

Why fine-tune?



Learn new
task or
domain



Reduced
time and
computation

How to fine-tune a model



Identify the
Model

How to fine-tune a model



Identify the
Model



Prepare
the Data

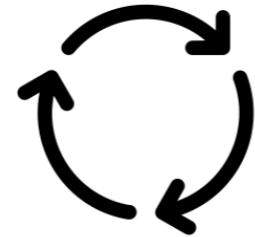
How to fine-tune a model



Identify the
Model



Prepare
the Data



Build
Trainer

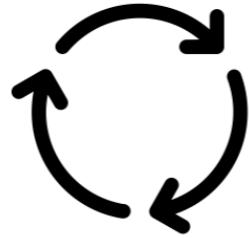
How to fine-tune a model



Identify the Model



Prepare the Data



Build Trainer



Train the Model

Identifying the model

```
from transformers import AutoModelForSequenceClassification  
  
model_name = "bert-base-cased"  
model = AutoModelForSequenceClassification.from_pretrained(model_name)
```

Preparing the dataset

```
# Identifying the model
from transformers import AutoModelForSequenceClassification

model_name = "bert-base-cased"
model = AutoModelForSequenceClassification.from_pretrained(model_name)
```

```
# Prepare the dataset
from transformers import AutoTokenizer

tokenizer = AutoTokenizer.from_pretrained(model_name)
dataset = dataset.map(
    lambda row: tokenizer(row['text']))
)
```

Create the training loop

```
from transformers import (
    Trainer,
    TrainingArguments
)

training_args = TrainingArguments(
    output_dir = "./results")
```

- `Trainer` used for fine-tuning a model
- `TrainingArguments` are parameters passed into the trainer object
- `output_dir` directory to store results

¹ https://huggingface.co/docs/transformers/main_classes/trainer

Training the model

```
training_args = TrainingArguments(  
    output_dir = "./results")  
  
trainer = Trainer(  
    model=model,  
    args=training_args,  
    train_dataset=dataset['train'],  
    eval_dataset=dataset['test'])  
)
```

Training the model

```
trainer.train()
```

```
local_path = "./fine_tuned_model"
```

```
trainer.save_model(local_path)
```

Final fine-tuned model

```
from transformers import pipeline

classifier = pipeline(task="text-classification", model=local_path)

classifier(dataset['text'])
```

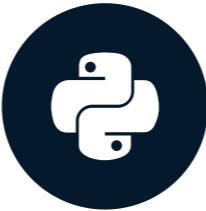
```
[{"Label": "Science", "Score":0.987}]
```

Let's practice!

WORKING WITH HUGGING FACE

Text generation

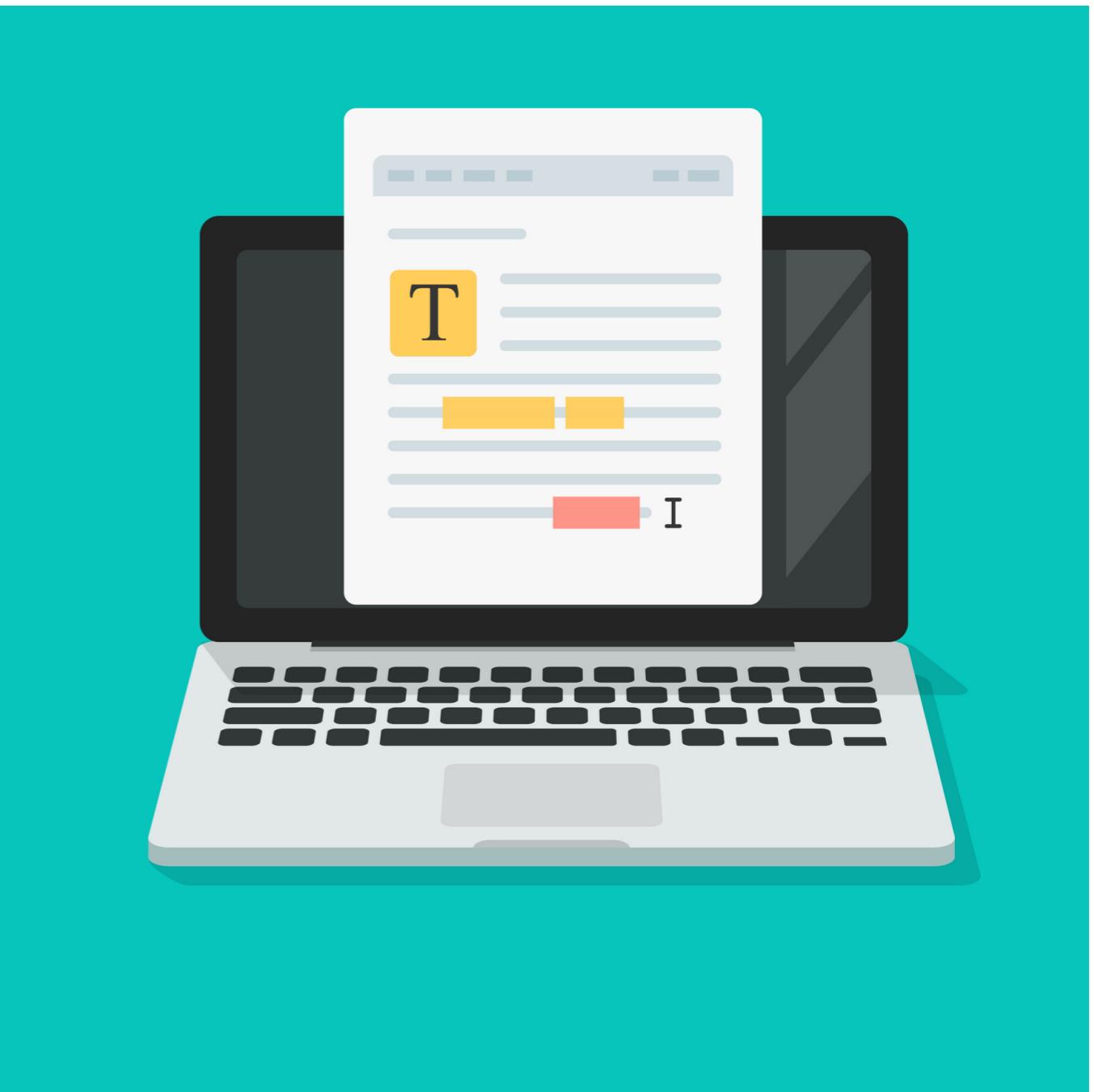
WORKING WITH HUGGING FACE



Jacob H. Marquez
Lead Data Engineer

What is text generation?

- Produces new text from input
- Useful for developing content
- Plays role in many ML tasks
- Can be fine-tuned for specific tasks



Inputs for text generation



Text input

Inputs for text generation



Text input



Microsoft

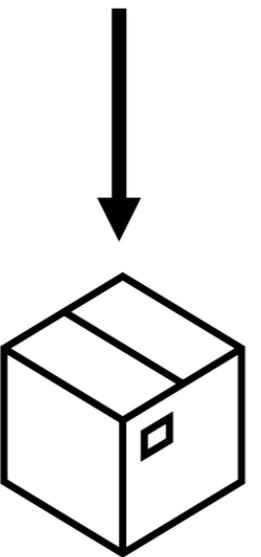
*Generative
Image-to-text
Transformer*

Image input

How do models generate text?

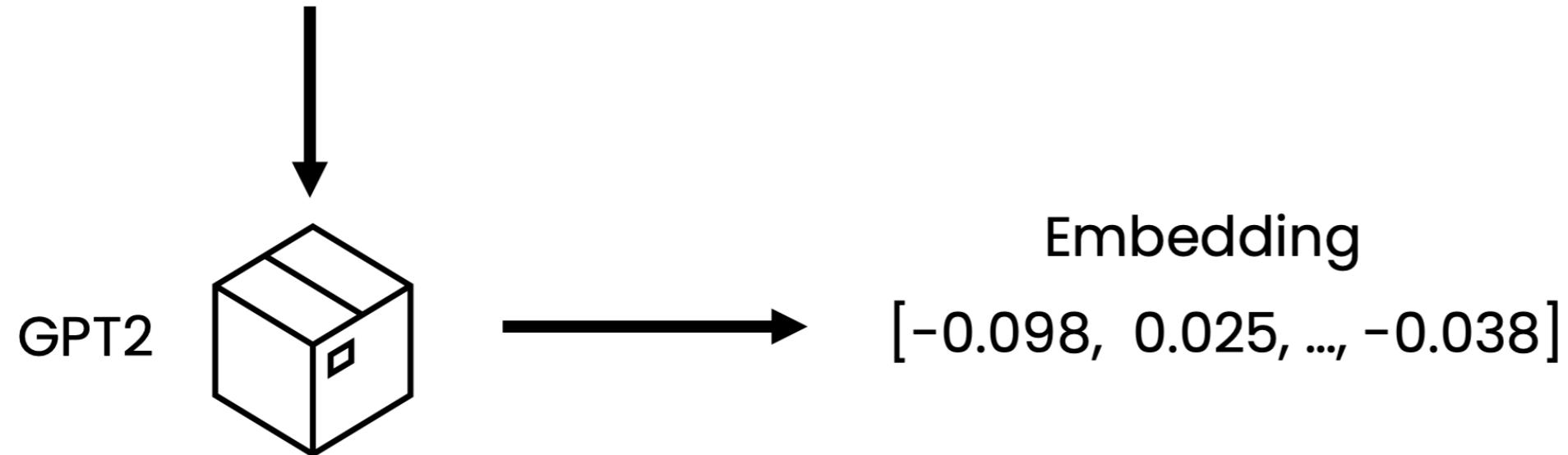
"pizza tastes the best with"

GPT2



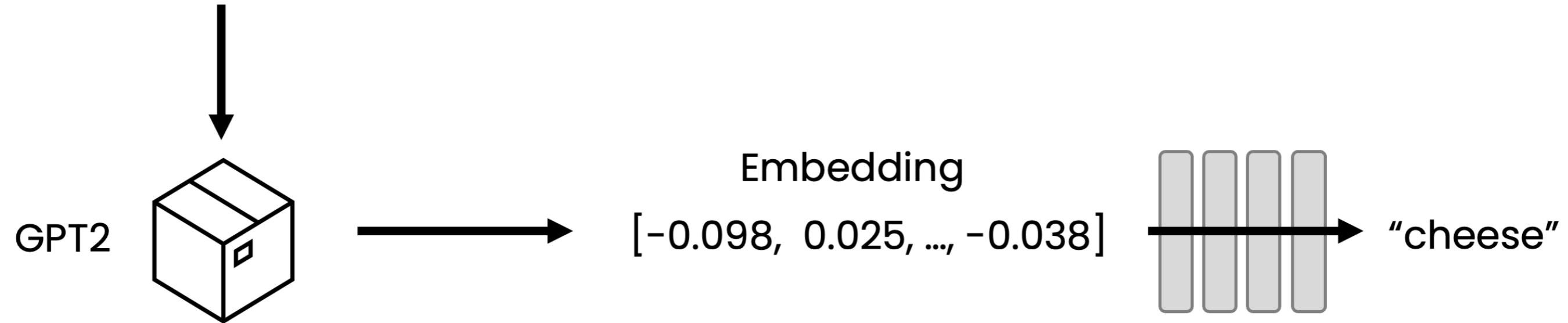
How do models generate text?

"pizza tastes the best with"



How do models generate text?

"pizza tastes the best with"



Generating text using Hugging Face

```
from transformers import pipeline  
  
generate_text = pipeline(task="text-generation")
```

```
from transformers import (  
    AutoTokenizer,  
    AutoModelForCausalLM  
)  
  
tokenizer = AutoTokenizer.from_pretrained("gpt2")  
model = AutoModelForCausalLM.from_pretrained("gpt2")
```

Generating text using Hugging Face

```
prompt = "Yogurt tastes better when topped with"
```

```
input_ids = tokenizer.encode(prompt, return_tensors="pt")
```

```
tensor([[ 101, 10930, 27390, 2102, 16958, 2488, 2043, 9370, 2007, 102]])
```

```
output = model.generate(input_ids, num_return_sequences=1)
```

```
generated_text = tokenizer.decode(output[0])
```

```
print(generated_text)
```

```
Yogurt tastes better when topped with a little bit of honey.
```

Generating text from an image

```
from transformers import (
    AutoProcessor,
    AutoModelForCausalLM)
from PIL import Image

proc = AutoProcessor.\n    from_pretrained(\n        "microsoft/git-base-coco")
model = AutoModelForCausalLM.\n    from_pretrained(\n        "microsoft/git-base-coco")
```

```
img = Image.open("jacob_original.jpeg")
```



Generating text from an image

```
pixel_values = proc(images=img, return_tensors="pt").pixel_values  
  
generated_ids = model.generate(pixel_values=pixel_values, max_length=50)  
  
generated_caption = proc.batch_decode(  
    generated_ids,  
    skip_special_tokens=True)  
  
print(generated_caption[0])
```

a man in a yellow beanie

Let's practice!

WORKING WITH HUGGING FACE

Embeddings

WORKING WITH HUGGING FACE



Jacob H. Marquez
Lead Data Engineer

What are embeddings?

Vector



$[-0.098, 0.025, \dots, -0.038]$

What are embeddings?

"What are embeddings?"



[-0.098, 0.025, ..., -0.038]

Vector

What are embeddings?

"What are embeddings?"

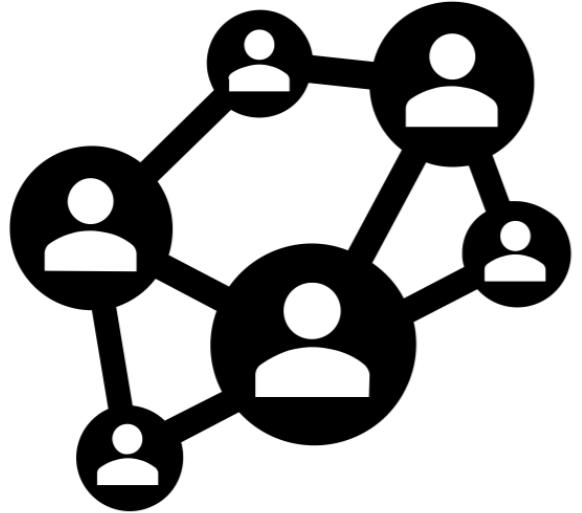


$[-0.098, 0.025, \dots, -0.038]$

Vector

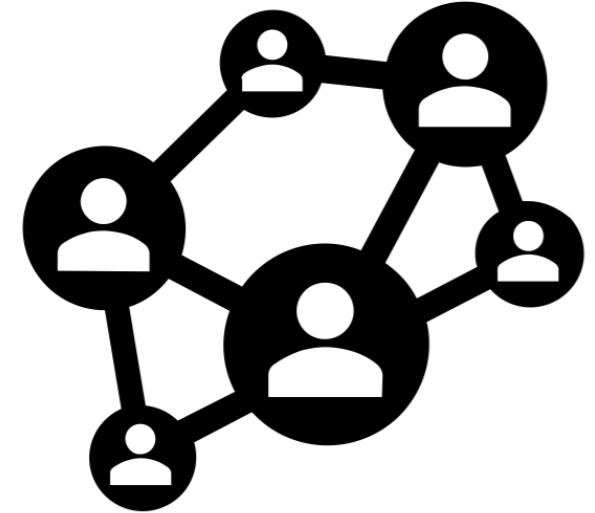
n-dimensions

Use cases for embeddings



Recommender
Systems

Use cases for embeddings

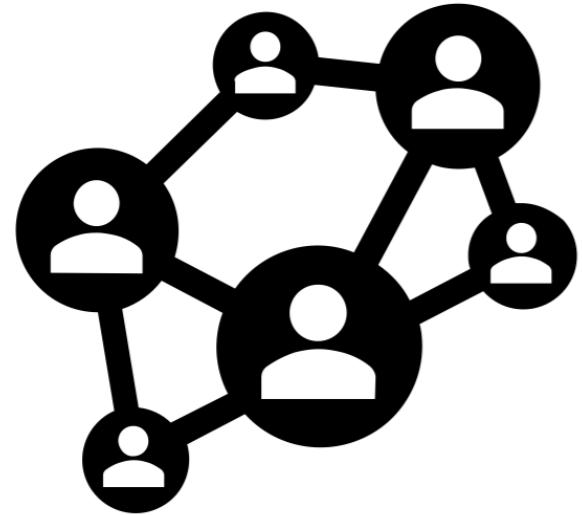


Recommender
Systems



Search

Use cases for embeddings



Recommender
Systems



Search



Fraud
Detection

Benefits

- Semantic understanding
- Features
- Improved generalization

Challenges

- Require lots of data
- Inherit biases
- Interpretability

Using embeddings in Hugging Face

```
pip install sentence_transformers
```

```
from sentence_transformers import SentenceTransformer  
  
model_name = "all-MiniLM-L6-v2"  
  
embedder = SentenceTransformer(model=model_name)
```

Using embeddings in Hugging Face

```
sentence = "What are embeddings?"  
  
embedding = embedder.encode([sentence])  
  
print(embedding)
```

```
array([[-1.73364417e-03, -8.55376422e-02, ...]])
```

```
print(embedding.shape)
```

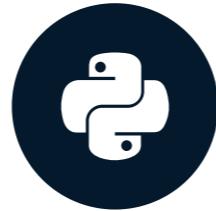
```
(384)
```

Let's practice!

WORKING WITH HUGGING FACE

Semantic search

WORKING WITH HUGGING FACE



Jacob H. Marquez
Lead Data Engineer

Keyword search

- Traditional approach
- Uses a set of keywords to find documents
- Best for simple or precise queries
- Less computationally intensive

search: monkey gene blockers

keywords: ["monkey", "gene", "blockers"]

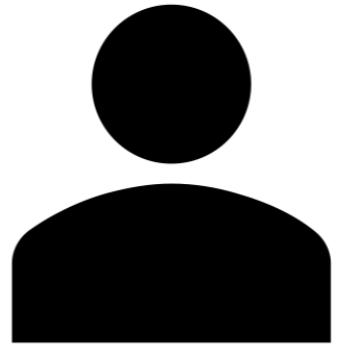
document: AG News Article

Gene Blocker Turns Monkeys Into Workaholics

- Study (Reuters) Reuters - Procrastinating monkeys were turned\into workaholics using a gene treatment to block a key brain\compound, U.S. researchers reported on Wednesday

Semantic search

- Understand intent and context
- Uses query input and other context
- Find content matching the meaning



User Info



Location



History

How does semantic search work?

- Vector search based on embeddings

How does semantic search work?

- Vector search based on embeddings



$[-0.098, 0.025, \dots, -0.038]$



$[0.768, -0.585, \dots, 0.238]$



$[-0.444, -0.105, \dots, 0.958]$

How does semantic search work?

- Vector search based on embeddings
- Compare embeddings using similarity measures
- More powerful with ambiguity and complexity

Similarity with cosine

Politics



Wildlife



Hip Hop



Similarity with cosine

Politics



$[-0.098, 0.025, \dots, -0.038]$

Wildlife



$[0.768, -0.585, \dots, 0.238]$

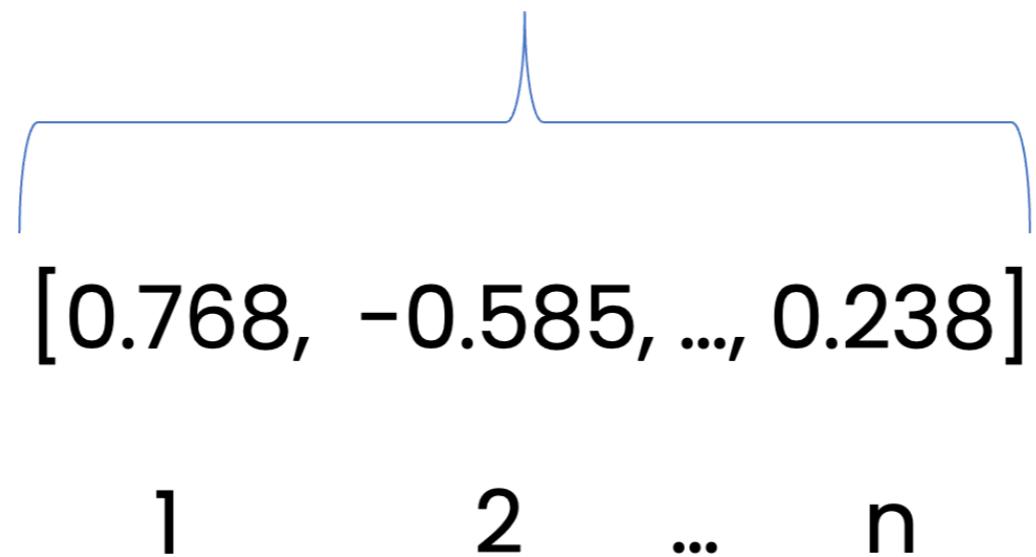
Hip Hop



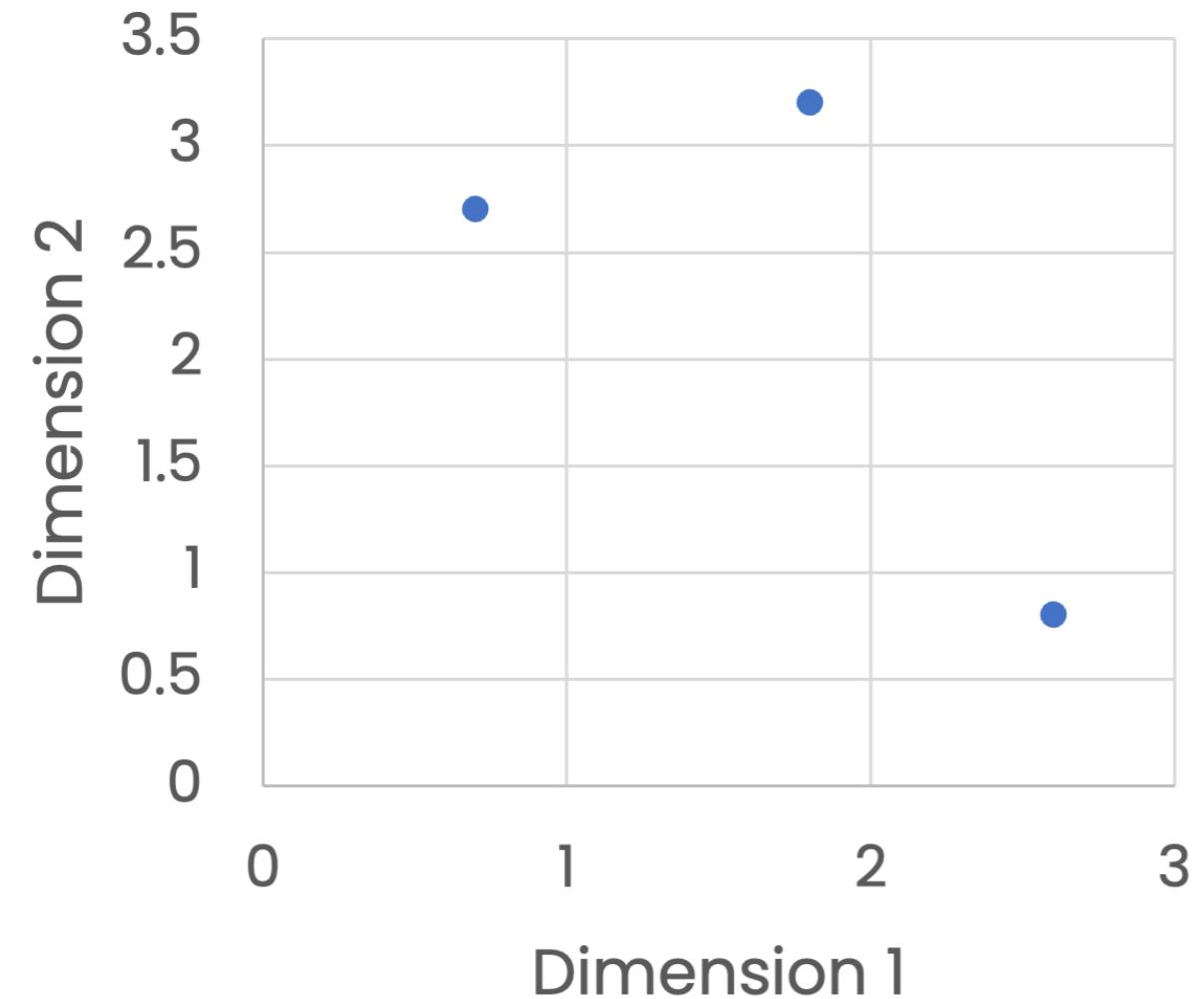
$[-0.444, -0.105, \dots, 0.958]$

Similarity with cosine

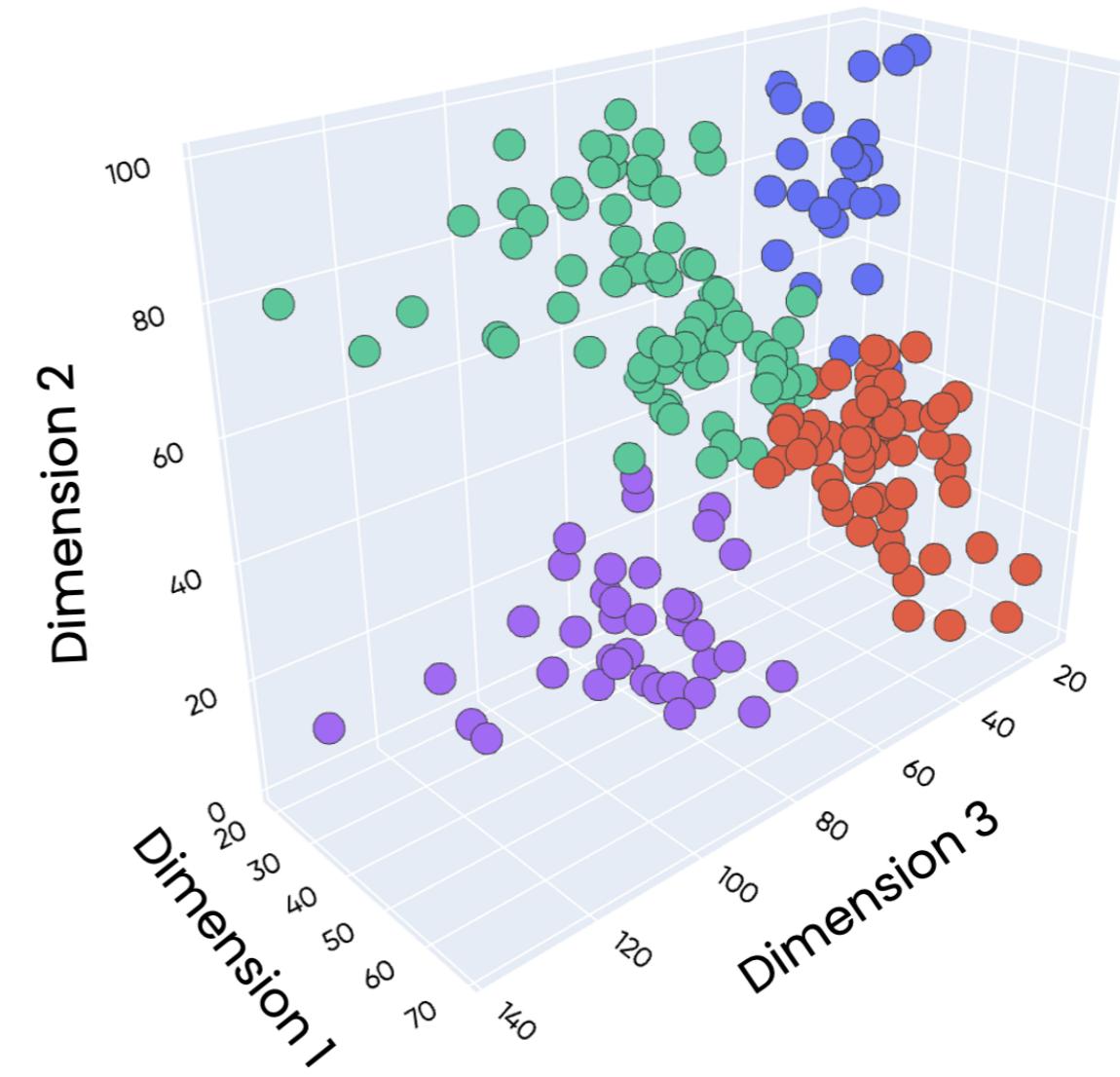
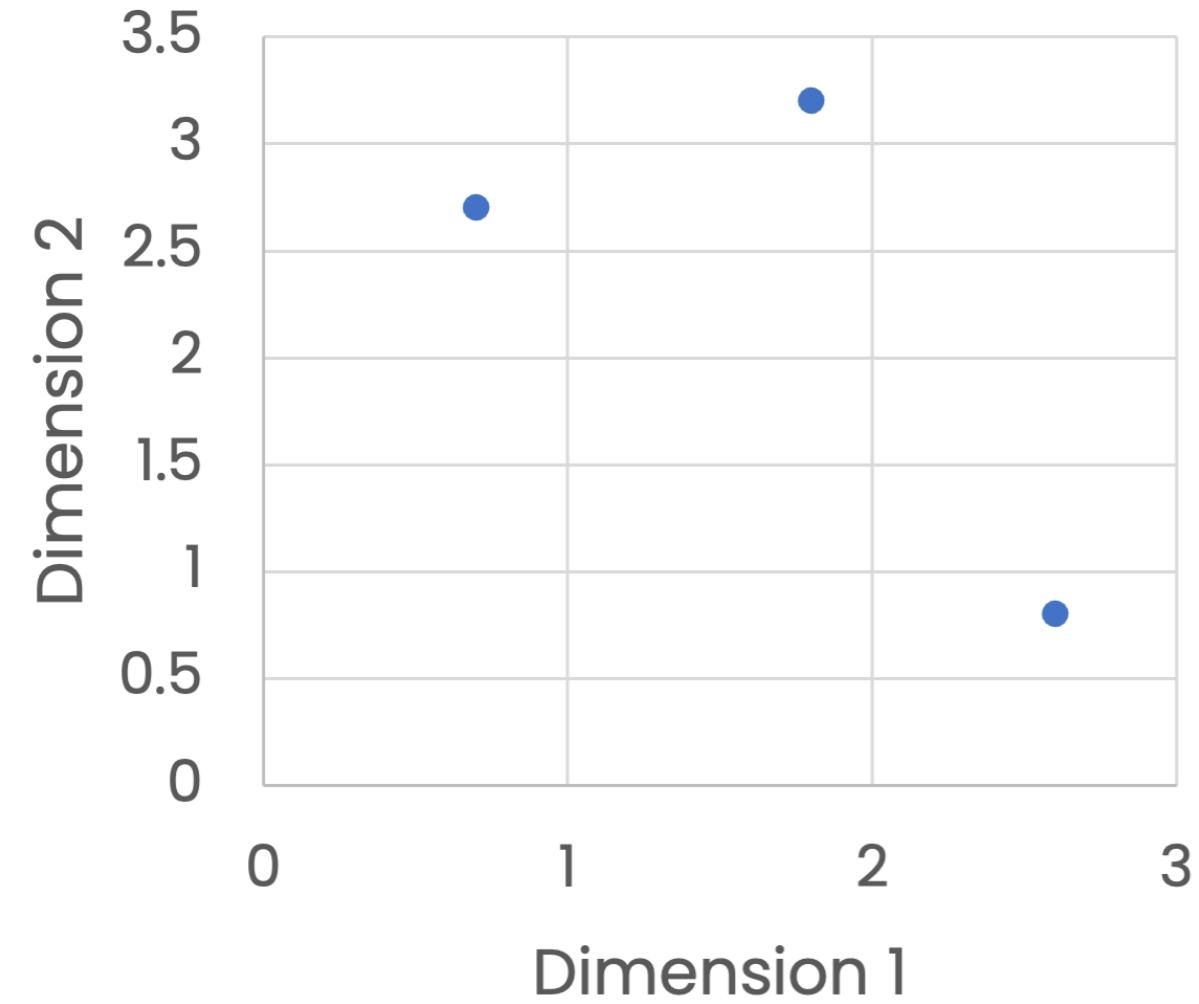
n-dimensions



Similarity with cosine

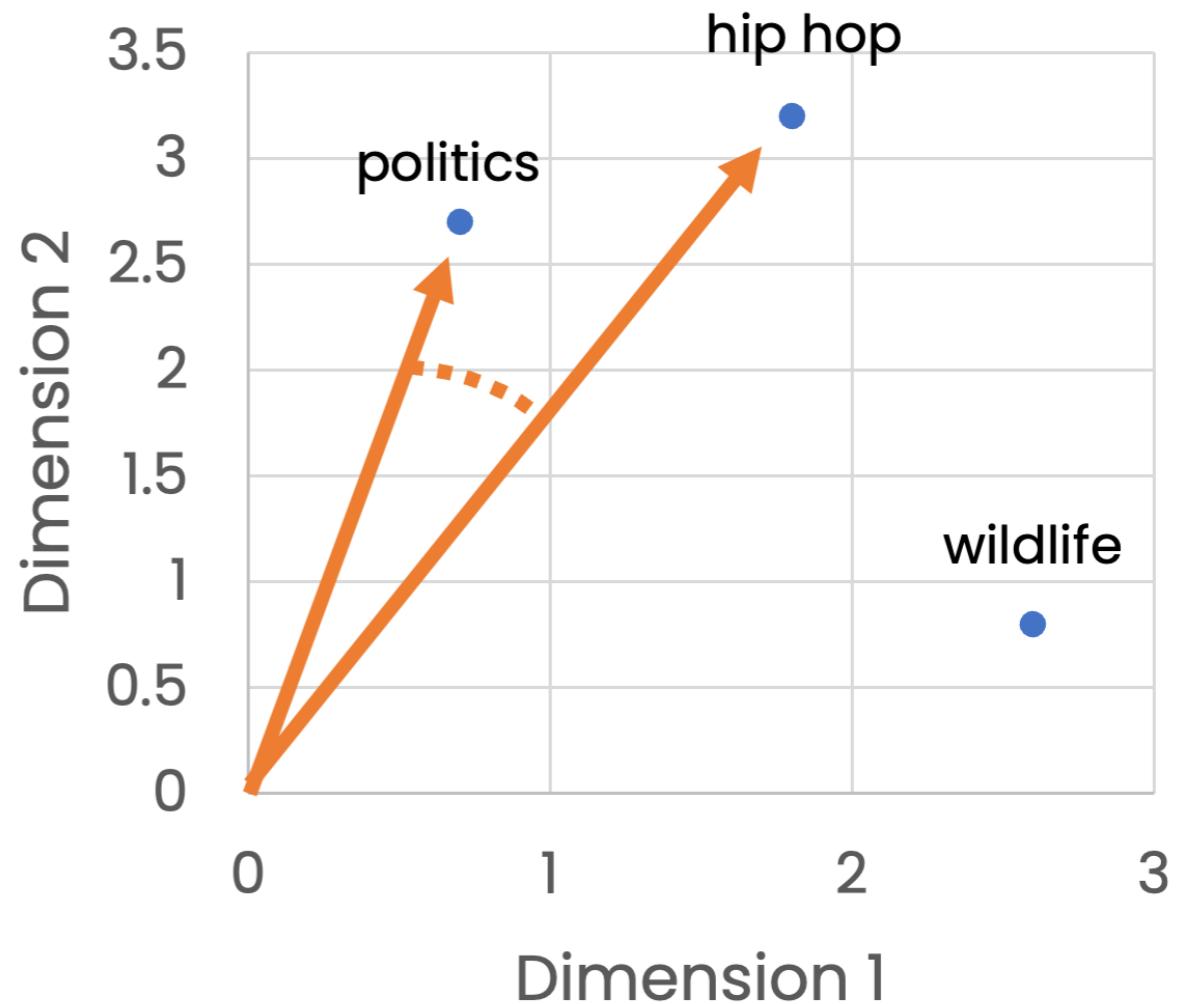


Similarity with cosine

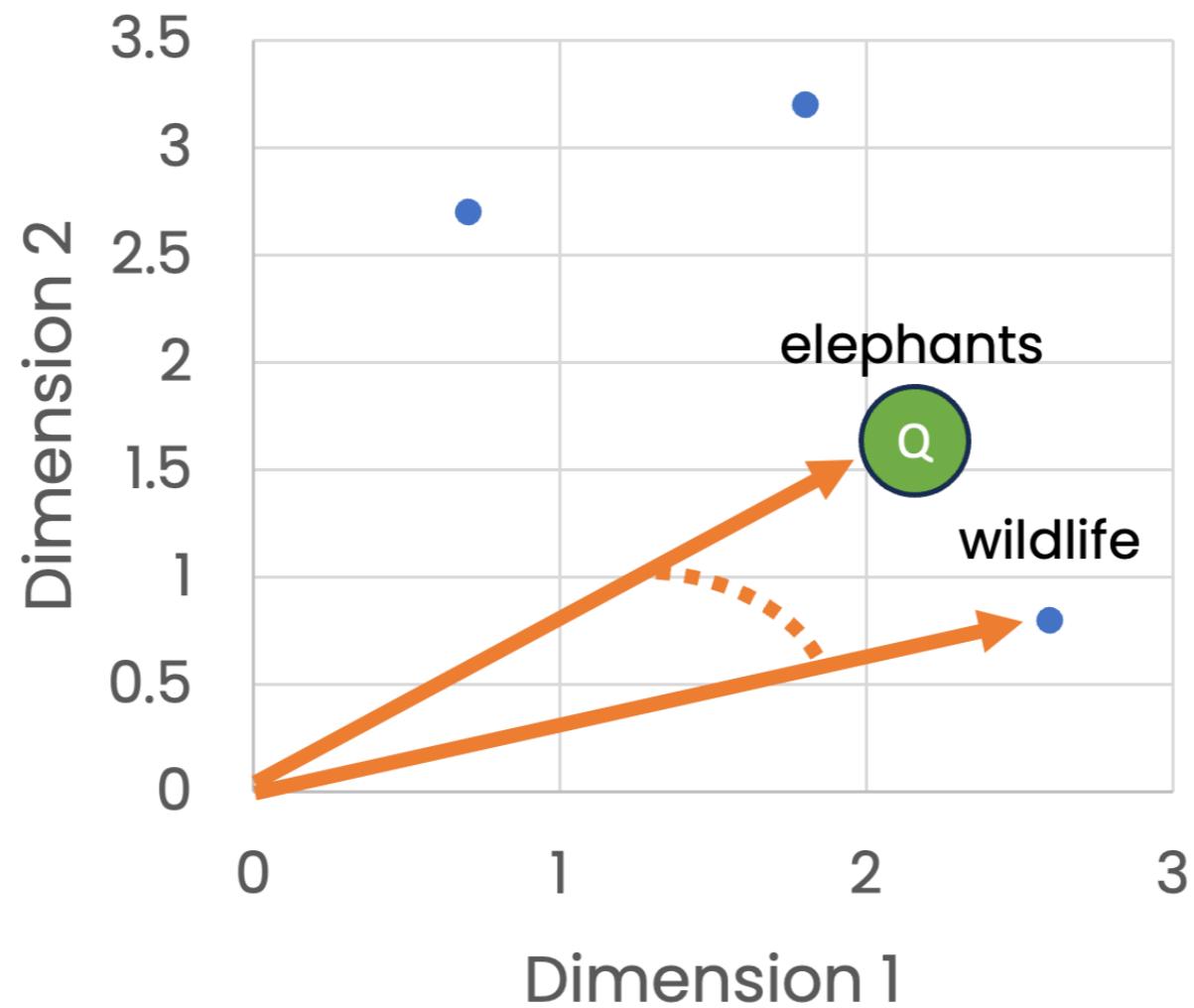


Cosine similarity

- Quantify distance between two points in n-dimensional space
- Calculated using the angle between two vectors for the points
- Similarity represented by dotted line
- Smaller = closer = similar



Searching with cosine



Semantic search with Hugging Face

```
from sentence_transformers import SentenceTransformers  
  
encoder = SentenceTransformers(model="all-MiniLM-L6-v2")  
  
document_embeddings = encoder.encode(documents)  
  
query = "What are the most recent wildlife articles?"  
  
query_embedding = encoder.encode([query])
```

Semantic search with Hugging Face

```
from sentence_transformers import util
```

```
hits = util.semantic_search(query_embedding, document_embeddings, top_k=2)
```

```
for hit in hits[0]:  
    print(documents[hit['corpus_id']], "(Score: {:.4f})".format(hit['score']))
```

"Migration of elephants across the ..." (Score: 0.9561)

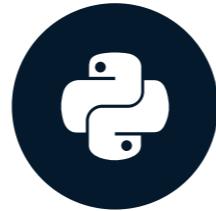
"What do birds do in the winter when ..." (Score: 0.8011)

Let's practice!

WORKING WITH HUGGING FACE

Congratulations

WORKING WITH HUGGING FACE



Jacob H. Marquez
Lead Data Engineer

Chapter 1 - Hugging Face

The screenshot shows the Hugging Face platform interface. On the left, there's a sidebar with options like '+ New', 'onthemarq' (a user profile), 'Profile', 'Inbox (0)', 'Settings', 'Get Pro', 'Organizations', 'Create New', 'Resources', and 'Hub guide'. The main area has a search bar 'Search models, datasets, spaces...' and navigation links for 'Models', 'Datasets', 'Spaces', 'Docs', 'Solutions', and 'Pricing'. The 'Following' section shows 0 items and filters for 'All', 'Models', 'Datasets', 'Spaces', 'Papers', and 'Collections', with sub-options for 'Community', 'Upvotes', and 'Likes'. A callout box highlights the 'Follow your favorite AI creators' feature with profiles for 'lewtn' (Expert in LLMs + RLHF), 'julien-c' (Person building this platform), and 'TheBloke' (Quantization of +2000 open source...). The 'Trending' section shows the top trending models over the last 7 days, including 'stabilityai/sdxl-turbo' (Text-to-Image, Up..., 224k, 777), 'stabilityai/stable-video-d...' (Updated 2 days ago, 71.8k, 1.13k), 'berkeley-nest/Starling-LM-...' (Text Generation, U..., 2.84k, 226), and 'Intel/neural-chat-7b-v3-1' (Text Generation, U..., 24k, 445).

Hugging Face Search models, datasets, spaces...

Models Datasets Spaces Docs Solutions Pricing

+ New

onthemarq

Profile Inbox (0) Settings Get Pro

Organizations Create New

Resources Hub guide

Following 0

All Models Datasets Spaces Papers Collections

Community Upvotes Likes

NEW Follow your favorite AI creators Refresh List

lewtn · Expert in LLMs + RLHF Follow

julien-c · Person building this platform Follow

TheBloke · Quantization of +2000 open sourc... Follow

Trending last 7 days

All Models Datasets Spaces

stabilityai/sdxl-turbo · Text-to-Image • Up... • 224k • 777

stabilityai/stable-video-d... · Updated 2 days ago • 71.8k • 1.13k

berkeley-nest/Starling-LM-... · Text Generation • U... • 2.84k • 226

Intel/neural-chat-7b-v3-1 · Text Generation • U... • 24k • 445

Chapter 2 - Pipelines

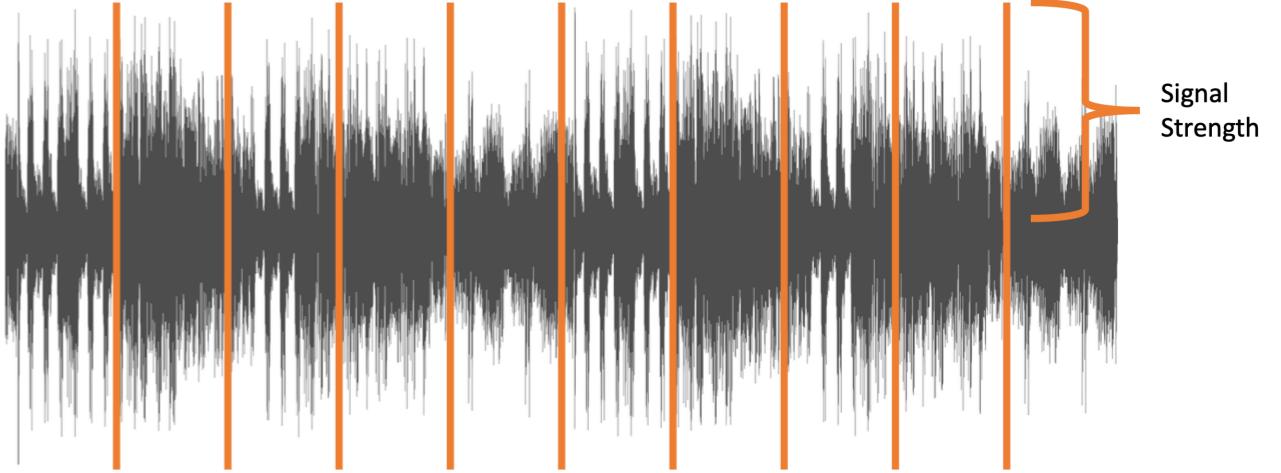
```
from transformers import pipeline

my_pipeline = pipeline(
    task="text-classification",
    model="distilbert-base-uncased-finetuned-sst-2-english")

input = "Hi, congrats on finishing the course!"

my_pipeline(input)
```

Chapter 3 - Pipelines with image and audio



Chapter 4 - Fine-tuning and advanced topics

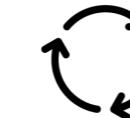
Fine-tuning



Identify the Model



Prepare the Data

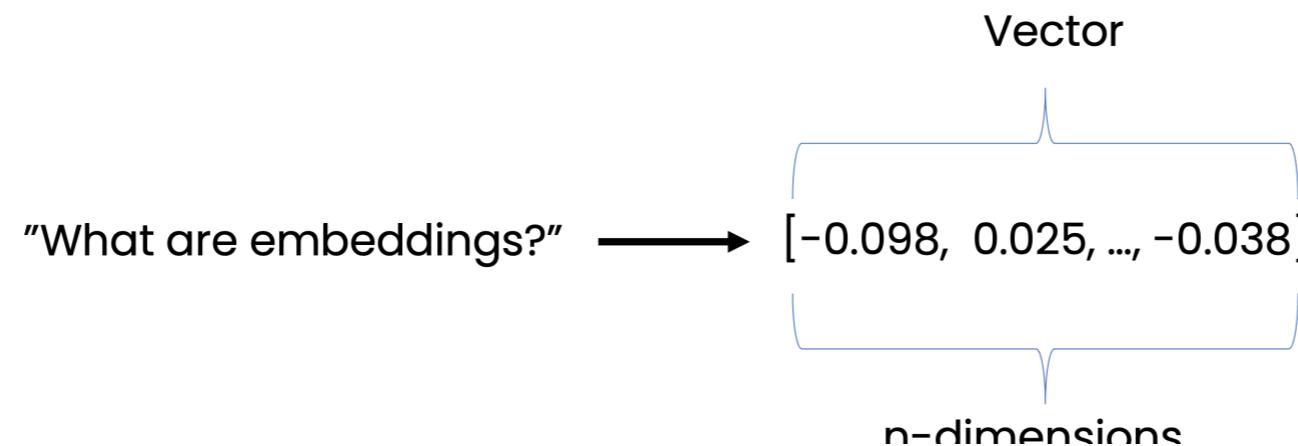


Build Trainer

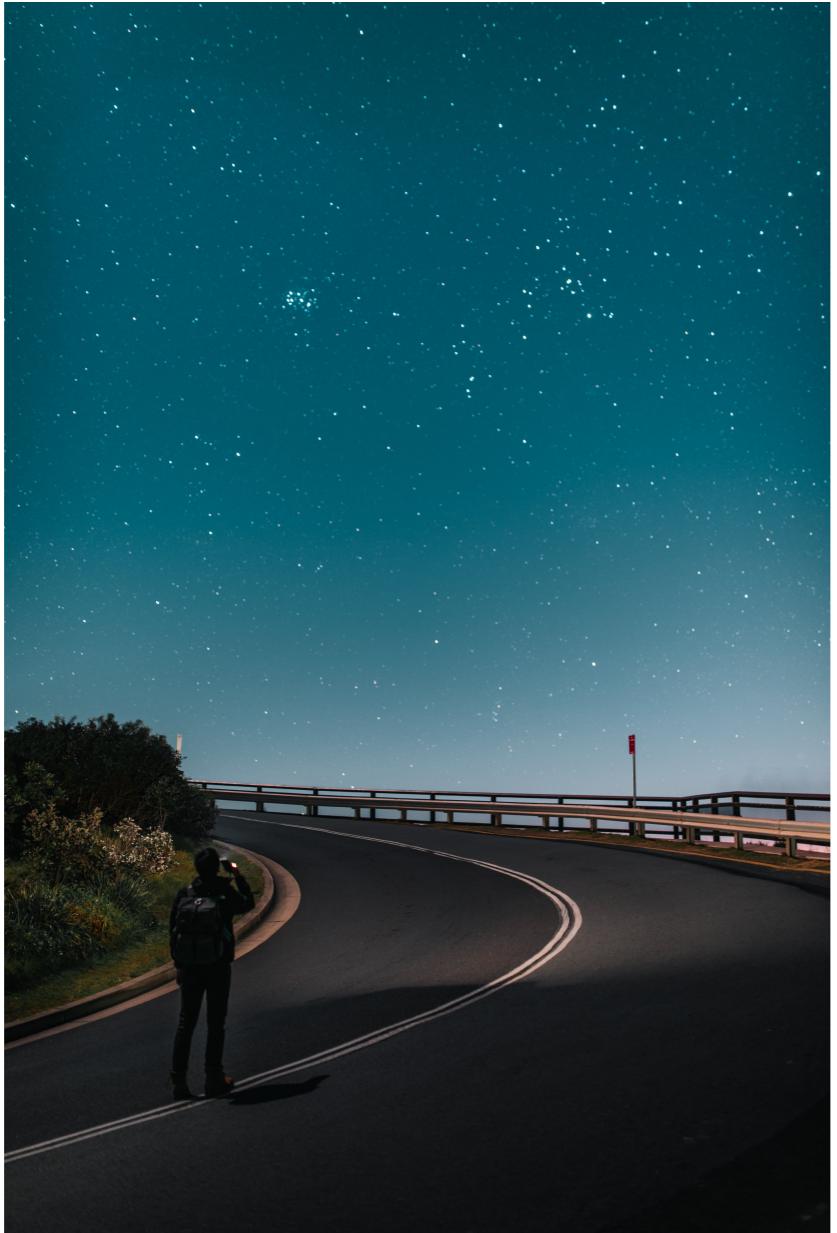


Train the Model

Embeddings



Next steps



- Explore the Hugging Face ecosystem
- Complex tasks, models
- Perform end-to-end fine-tuning

Congratulations!

WORKING WITH HUGGING FACE