

# AI detector with frontend interface

1. Uses a technology called DistilBERT - a compact AI model that understands language
2. This model has been trained to recognize differences between human and AI writing styles
3. It's like a language detective that spots subtle patterns most people miss
4. Similar to how you might recognize a friend's writing style, but much more precise

Trained on

[https://huggingface.co/datasets/NabeelShar/ai\\_and\\_human\\_text](https://huggingface.co/datasets/NabeelShar/ai_and_human_text)

Datasets: NabeelShar/ai\_and\_human\_text like 0

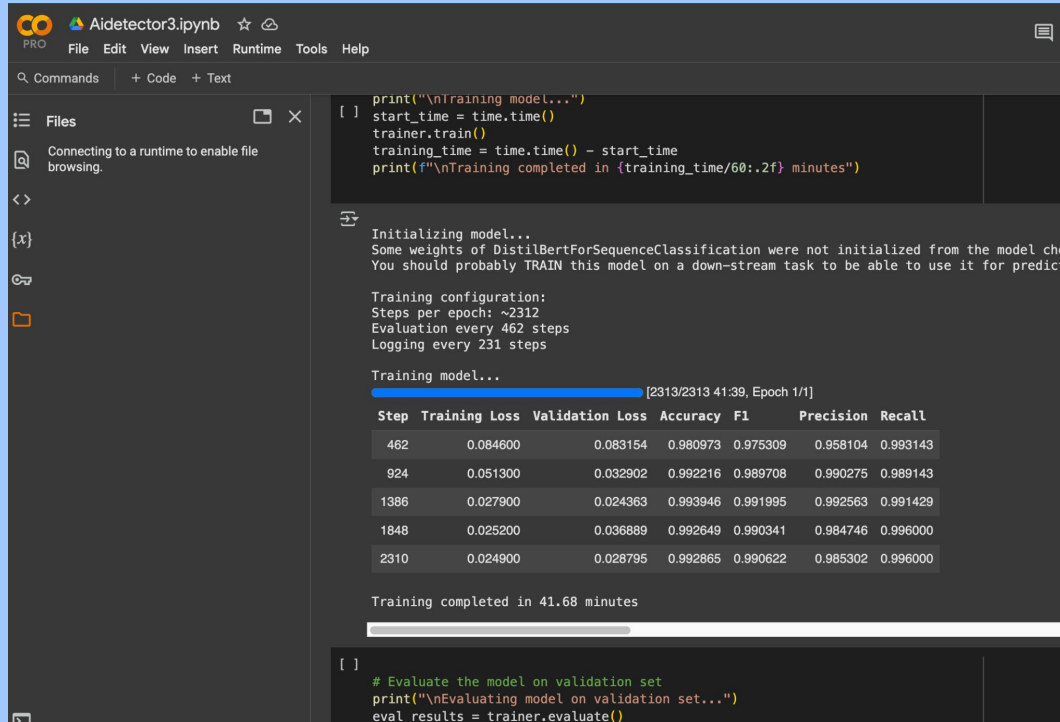
Split (1)  
train · 46.2k rows

Search this dataset

text	generated	prompt_name
string · lengths	int64	string · classes
48 18.3k	0 1	15 values
Cars. Cars have been around since they became famous in the 1900s, when Henry Ford created and built the first...	0	Car-free cities
Transportation is a large necessity in most countries worldwide. With no doubt, cars, buses, and other means...	0	Car-free cities
"America's love affair with it's vehicles seems to be cooling" says Elisabeth rosenthal. To understand...	0	Car-free cities
How often do you ride in a car? Do you drive a one or any other motor vehicle to work? The store? To the...	0	Car-free cities
Cars are a wonderful thing. They are perhaps one of the worlds greatest advancements and technologies. Cars ge...	0	Car-free cities
The electrol college system is an unfair system, people don't have the right to select their own president,...	0	Does the electoral college work?
Dear state senator, It is the utmost respect that I ask for the method for presidential election be changed...	0	Does the electoral college work?
Fellow citizens, cars have become a major role in our daily lives. They have their many excellent uses,...	0	Car-free cities
"It's official: The electoral college is unfair, outdated, and irrational" Plumer, Source 2. Many do no...	0	Does the electoral college work?

Getting a quality dataset that had good examples, lots of data, and thus prevented overfitting took three tries. This was my third dataset and script.

# Model trained on google Collab



```
[ ] print("\ntraining model...")
start_time = time.time()
trainer.train()
training_time = time.time() - start_time
print(f"\nTraining completed in {training_time/60:.2f} minutes")
```

Initializing model...  
Some weights of DistilBertForSequenceClassification were not initialized from the model che  
You should probably TRAIN this model on a down-stream task to be able to use it for predict

Training configuration:  
Steps per epoch: ~2312  
Evaluation every 462 steps  
Logging every 231 steps

Training model... [2313/2313 41:39, Epoch 1/1]

Step	Training Loss	Validation Loss	Accuracy	F1	Precision	Recall
462	0.084600	0.083154	0.980973	0.975309	0.958104	0.993143
924	0.051300	0.032902	0.992216	0.989708	0.990275	0.989143
1386	0.027900	0.024363	0.993946	0.991995	0.992563	0.991429
1848	0.025200	0.036889	0.992649	0.990341	0.984746	0.996000
2310	0.024900	0.028795	0.992865	0.990622	0.985302	0.996000

Training completed in 41.68 minutes

```
[ ] # Evaluate the model on validation set
print("\nEvaluating model on validation set...")
eval_results = trainer.evaluate()
```

The dataset is imported using a link of the hugging face dataset and opened up, processed, and the loss and other metrics are seen here.

- The model doesn't read words like we do - it breaks text into smaller pieces called "tokens"
- For example, "unbelievable" becomes three pieces: "un" + "believe" + "able"
- This helps it understand parts of words and handle words it hasn't seen before
- Every token gets converted to a number that the AI can process

# Tokenization

## <> Tokenization Visualization

See how the DistilBERT tokenizer processes your text

### Text Tokenization

3 tokens / 2 words

#### # Complete Token Sequence (with special tokens)

[CLS]	testing	token	##ization	[SEP]
101	5604	19204	3989	102

#### How DistilBERT Tokenization Works:

1. Special tokens like [CLS] and [SEP] are added at the beginning and end
2. Words are split into subwords (tokens) based on frequency
3. Tokens starting with "##" are continuations of the previous word
4. Each token is assigned a numeric ID from the vocabulary
5. These token IDs are what the model actually processes

Reset

The AI can only look at 512 tokens (roughly 300-400 words) at once  
For longer texts, we use a "sliding window" approach:

- Break the text into overlapping chunks
- Analyze each chunk separately
- Combine the results to get the final answer

Like reading a book by examining overlapping pages rather than the whole book at once

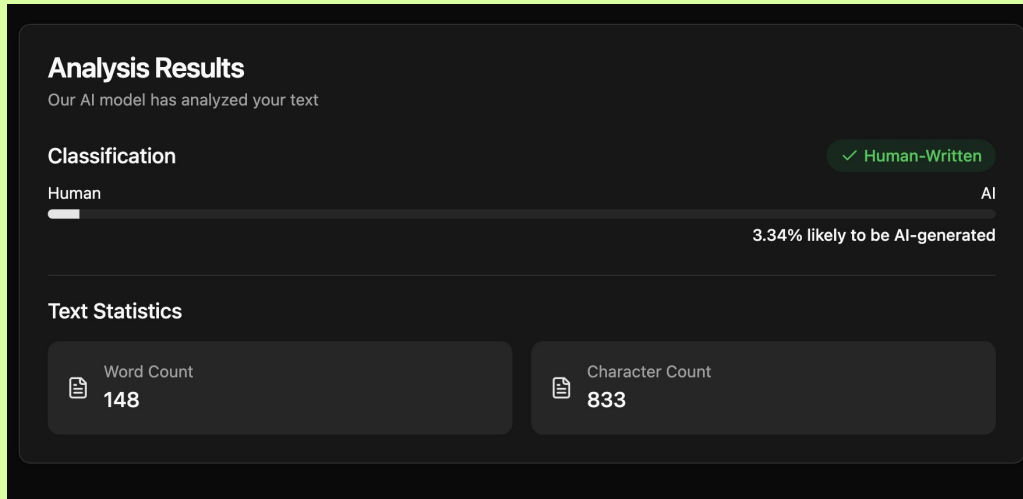
The overlaps are 256 tokens

This is due to the nature of the model -you must input an exact length.

Shorter ones have tokens added to them.

# 512 token input and larger inputs

The system doesn't just give a yes/no answer - it tells you how confident it is  
A result might be "85% likely to be AI-generated"  
I adjust this confidence using a "temperature" setting to make it more reliable  
Higher confidence means the AI is more certain about its decision





Flask handles requests from users  
When you submit text, it:

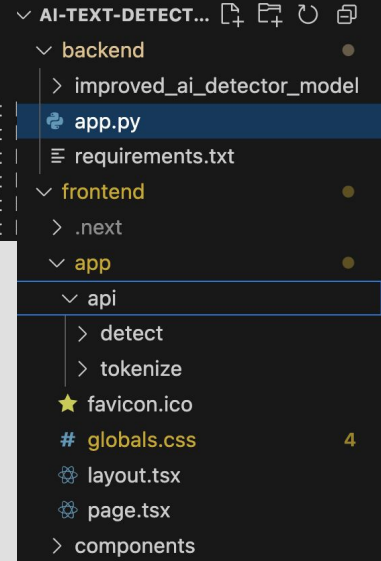
Prepares the text for the AI model  
Runs the model to get a prediction  
Calculates confidence scores  
Sends results back to the website

All the heavy AI processing happens here

There is also the tokenization route and logic

# Backend

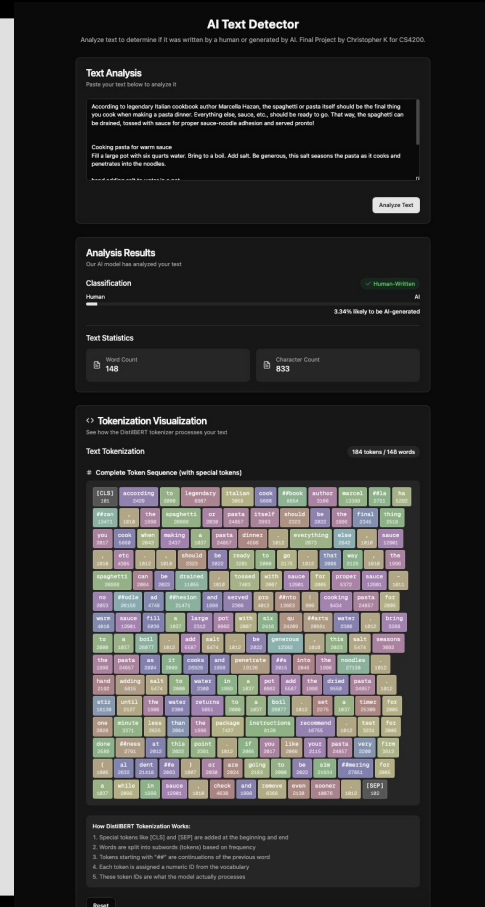
```
(base) christopher@b01-aruba-authenticated-10-110-200-80 backend % python3
Loading improved model...
Model loaded successfully on cpu!
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with watchdog (fsevents)
Loading improved model...
Model loaded successfully on cpu!
* Debugger is active!
* Debugger PIN: 105-220-697
127.0.0.1 - - [15/Apr/2025 11:53:20] "POST /api/detect"
127.0.0.1 - - [15/Apr/2025 11:53:31] "POST /api/detect"
127.0.0.1 - - [15/Apr/2025 11:53:31] "POST /api/detect"
127.0.0.1 - - [15/Apr/2025 11:53:51] "POST /api/detect"
127.0.0.1 - - [15/Apr/2025 11:53:53] "POST /api/detect"
127.0.0.1 - - [15/Apr/2025 11:53:57] "POST /api/detect"
```



## Built using modern web technology (Next.js)

- Features a simple text input box where you paste your text
- Shows results with easy-to-understand visuals
- Includes educational sections that explain how AI detection works
- Api routing in the api/ folder handles forwarding the requests to the flask backend

# Frontend



Setting up the AI part:

Install Python on your computer

Download the project files

Open a command window and type: `cd backend`

Install required programs: `pip install -r requirements.txt`

Start the AI server: `python app.py`

Setting up the website:

Install Node.js on your computer

Open a new command window and type: `cd frontend`

Install website components: `npm install`

Start the website: `npm run dev`

Open your web browser to: `http://localhost:4000`

# Setup

<http://localhost:4000>

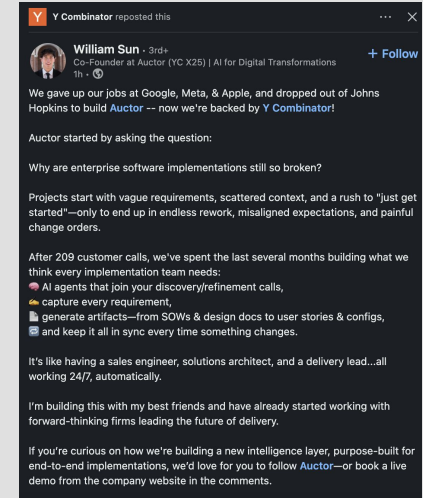
<https://github.com/christopherk26/ai-text-detector>

Human text:

<https://feelgoodfoodie.net/recipe/how-to-cook-pasta/>

AI text: (found this on linkedin - the best place to find ai generated content)

<https://www.linkedin.com/feed/update/urn:li:activity:7317969983924293635/>



# Link and demo

- Learned about LLMs, tokenization, NN's
- Realized that quality ai detectors are hard to make in general and are easy to trick
- Ai detectors in general are not very good (tried other ones online)
- What does this mean for the state of the internet (dead internet theory?)
- Incorporating several models together (what if we try using an llm and give it text and ask it to decide?)

# Takeaways

If you're curious on how we're building a new intelligence layer, purpose-built for end-to-end implementations, we'd love for you to follow Auctor—or book a live demo from the company website in the comments.

Oh—and meet our fifth cofounder (Chief Bark Officer 🐶) in the photo!

give me a percentage rating that you think this is ai generated

I'd give this post about a **35-45%** likelihood of being AI-generated. Here's why:

#### Reasons it *might* be AI-generated:

- **Polished, structured language:** The writing is clean, confident, and flows logically — something AI is good at.
- **Startup-style tone:** The post nails the typical startup announcement voice, with crisp formatting (emojis, bullet points, punchy intro), which AI <sup>tor</sup> often mimic well.

# Thanks!



Christopher Kurdoghlian

Spring 2025 CS2400 AI