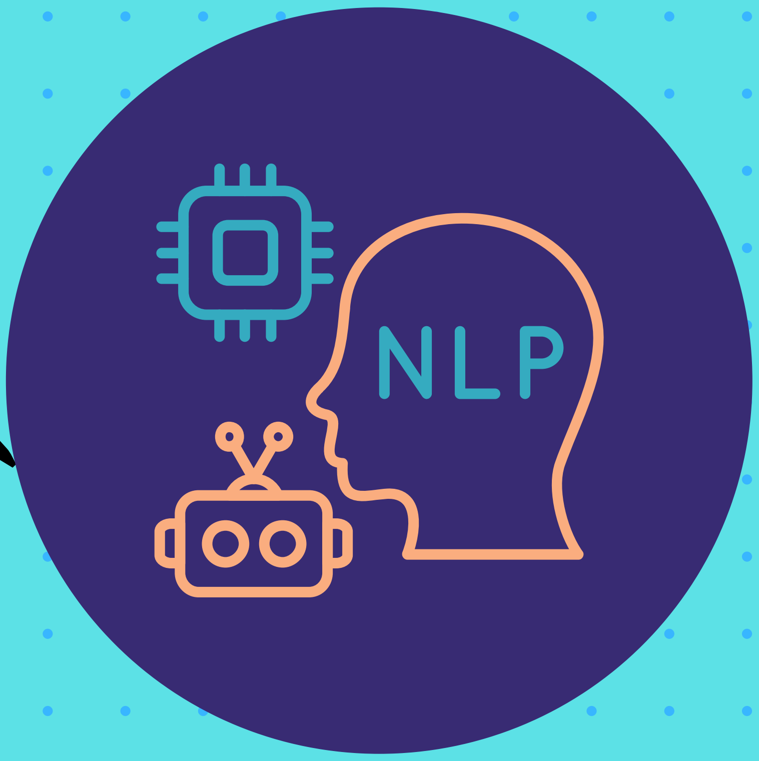
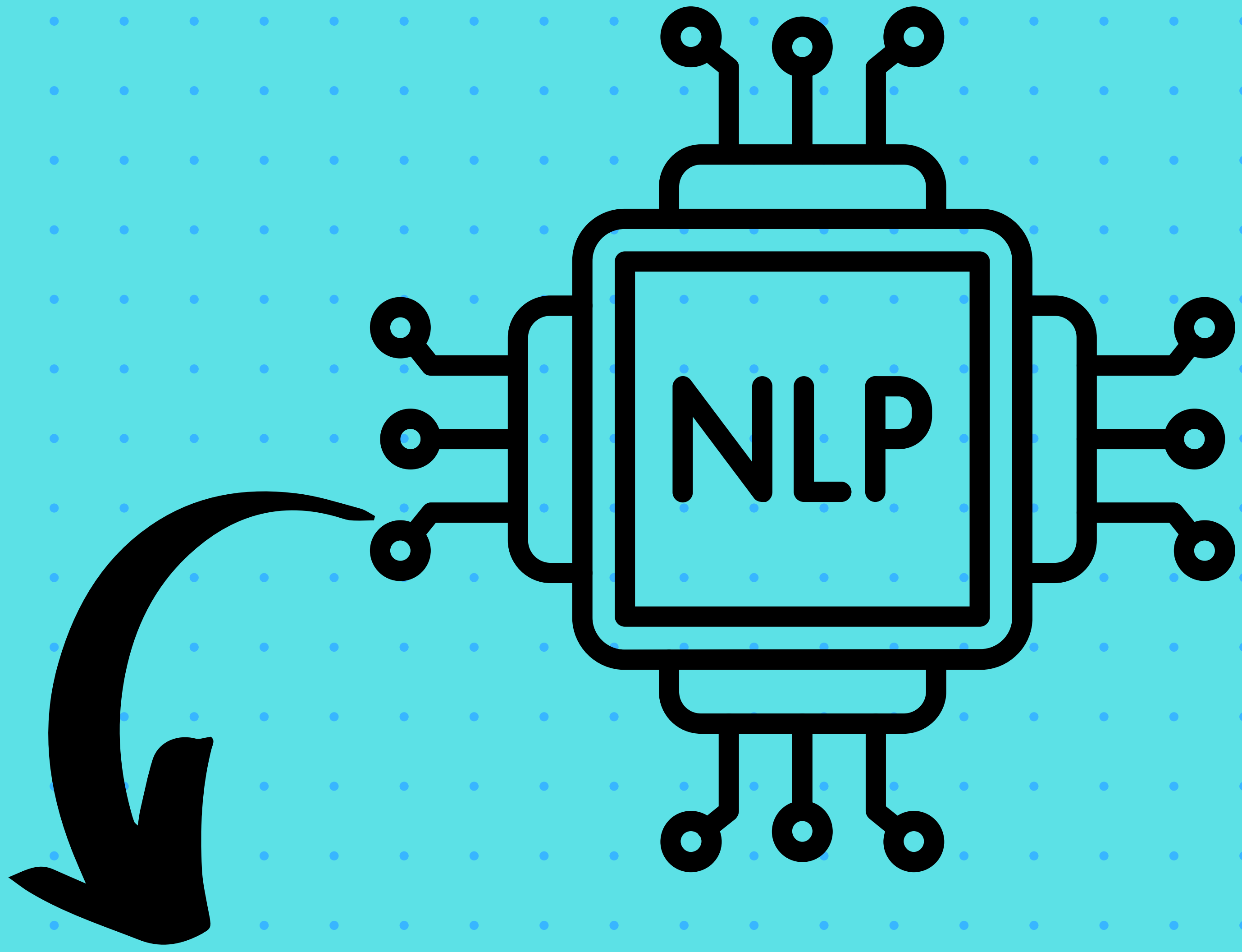
Build an NLP pipeline in Python



Hands-On NLP: From Raw Text to Sentiment in Minutes

Step-by-step code example: Tokenization, POS tagging, NER, and Sentiment Analysis

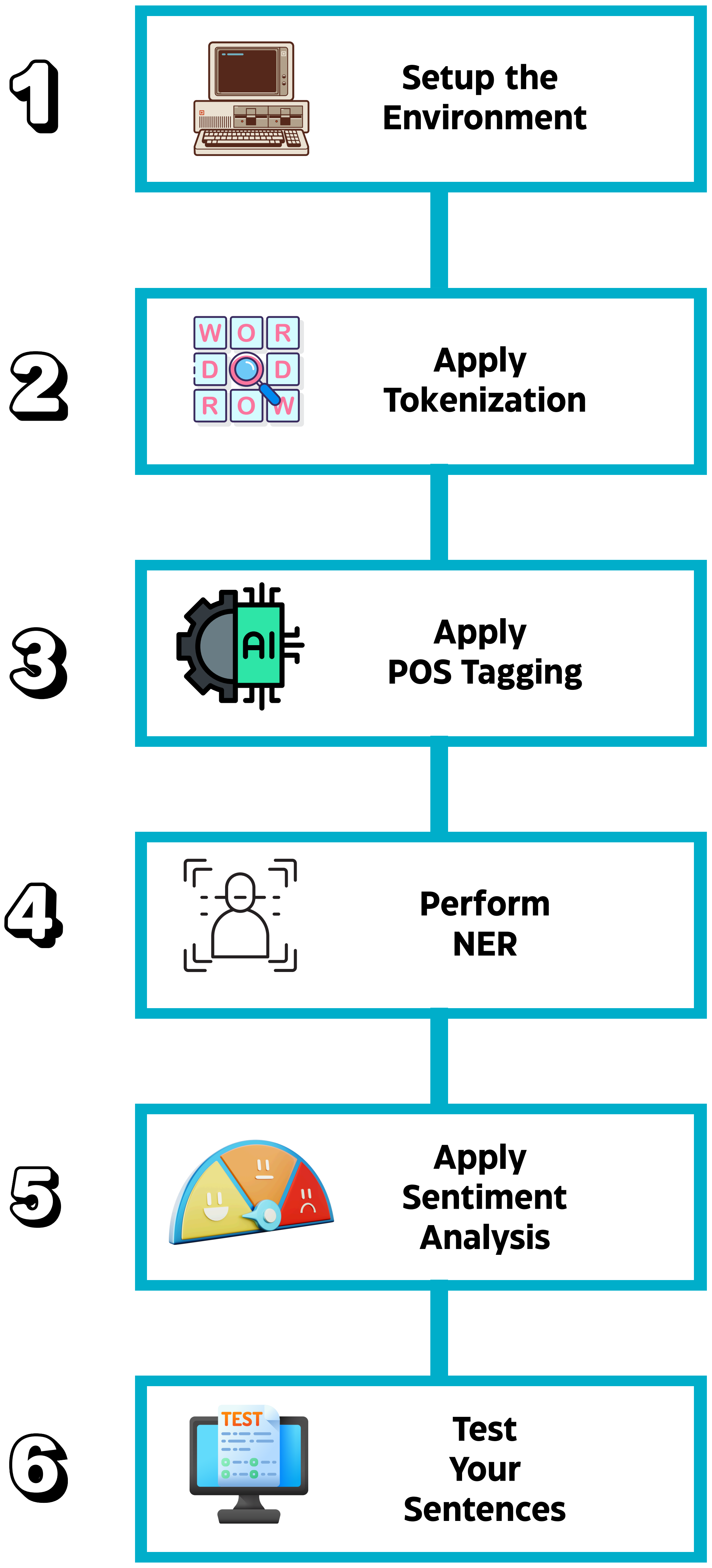
@Dzan Dedukic



What you’ll learn:

1. Workflow overview
2. Setup the environment
3. Preprocess Text with Tokenization
4. Apply POS Tagging
5. Perform NER
6. Apply Sentiment Analysis
7. Test the System

1. 🚀 Workflow overview



2. Setup the Environment

python™

Setup the Environment

```
pip install nltk spacy transformers
```

Install necessary libraries for tokenization, POS tagging, NER, and sentiment analysis.

✓ **NLTK (Natural Language Toolkit)**
🔹 Used for basic NLP tasks like tokenization, stemming, POS tagging, and parsing.

✓ **spaCy**
🔹 Designed for efficient NLP tasks like named entity recognition (NER), dependency parsing, and text classification.

✓ **Transformers (Hugging Face)**
🔹 Provides pretrained models for advanced NLP tasks such as text generation, sentiment classification, and translation.

3. Apply Tokenization

Apply Tokenization

```
import nltk
nltk.download('punkt')
nltk.tokenize import word_tokenize

text = "Hello world. How are you?"
tokens = word_tokenize(text)
print(tokens)
```

```
#Output ['Hello', 'world', '.', 'How', 'are', 'you', '?']
```

3. Apply Tokenization

Apply Tokenization

What's happening here?

1. **Download punkt** (so NLTK knows how to split text).
2. **Tokenize your text into words or sentences**.
3. **Use the tokens in your AI or NLP project!**

Why is tokenization necessary?

- Turns raw text into data you can process: Tokenization splits big text blocks into sentences or words.
- Prepares text for analysis: Most NLP tasks—like counting words, finding keywords, or building models—need text in small, manageable pieces.
- Foundation for all NLP: Without tokenization, your model can't "understand" or work with language.

4. Apply POS Tagging

Apply POS Tagging

```
nltk.download('averaged_perceptron_tagger_eng')
# Apply POS tagging
tagged_tokens = nltk.pos_tag(tokens)
print(tagged_tokens)
```

```
# Output:
[('Hello', 'NNP'), ('world', 'NN'), ('.', '.'), ('How',
'WRB'), ('are', 'VBP'), ('you', 'PRP'), ('?', '.')]
```

NNP — Proper Noun (e.g., names, “Hello” here)
NN — Noun, Singular (“world”)
. — Punctuation (period, question mark, etc.)
WRB — Wh-adverb (question words like “how”, “when”, “where”, “why”)
VBP — Verb, present tense, plural (“are”)
PRP — Personal pronoun (“you”)

4. Apply POS Tagging

Apply POS Tagging

What's happening here?

After tokenizing the text, you apply Part-of-Speech (POS) tagging using NLTK.
This step labels each word (token) with its grammatical role—such as noun, verb, adjective, etc.

Why does this matter?
POS tagging helps the system understand the structure of a sentence.

It's a foundation for advanced NLP tasks like sentiment analysis, question answering, and text generation.

The **averaged_perceptron_tagger_eng** is a pre-trained model in NLTK that assigns part-of-speech (POS) tags to each word in English text, helping identify nouns, verbs, adjectives, and more.

5. Perform NER

Perform
NER

```
#Change text variable to
text = "Apple was founded by Steve Jobs and
Steve Wozniak in California in 1976."
import spacy
# Load the pretrained model for NER
nlp = spacy.load("en_core_web_sm")
# Process the text with NER
doc = nlp(text)
# Extract entities
for ent in doc.ents:
    print(ent.text, ent.label_)
```

```
#Output:
Apple ORG
Steve Jobs PERSON
Steve Wozniak PERSON
California GPE
1976 DATE
```

5. Perform NER

Perform NER

What’s happening here?
(Named Entity Recognition – NER)

- After tagging words, we use spaCy’s pre-trained model to find and label important entities in the text—like people, companies, or places.
- The code processes your text and prints out each recognized entity along with its type (e.g., PERSON, ORG, GPE).

Why does it matter?
NER helps you pull out key information for tasks like data extraction, search, and automatic reporting.

If you get an error like OSError: [E050] Can't find model 'en_core_web_sm', you need to install the spaCy English model.

Open your terminal and run:

```
python -m spacy download en_core_web_sm
```

6. Apply Sentiment Analysis

Apply Sentiment Analysis

```
from transformers import pipeline

# Initialize sentiment analysis pipeline
sentiment_analyzer = pipeline('sentiment-
analysis')

# Analyze the sentiment of the text
result = sentiment_analyzer(text)
print(result)
```

```
#Expected output based on
distilbert/distilbert-base-uncased-finetuned-sst-2-
english and revision 714eb0f:

[{'label': 'POSITIVE', 'score': 0.9771140217781067}]
```

6. Apply Sentiment Analysis

Apply Sentiment Analysis

What's happening here? (Sentiment Analysis)

- After extracting entities, you use Hugging Face's transformers library to analyze the emotional tone of the text.
- The pipeline automatically classifies your text as positive, negative, or neutral.

Why does it matter?
Sentiment analysis reveals whether the message is happy, angry, neutral, etc.—helpful for understanding opinions, feedback, or trends.

Note:
The transformers sentiment analysis pipeline requires either PyTorch (torch) or TensorFlow to run models. If you get an error about missing frameworks, install PyTorch with:
pip install torch

7. Do Some Tests

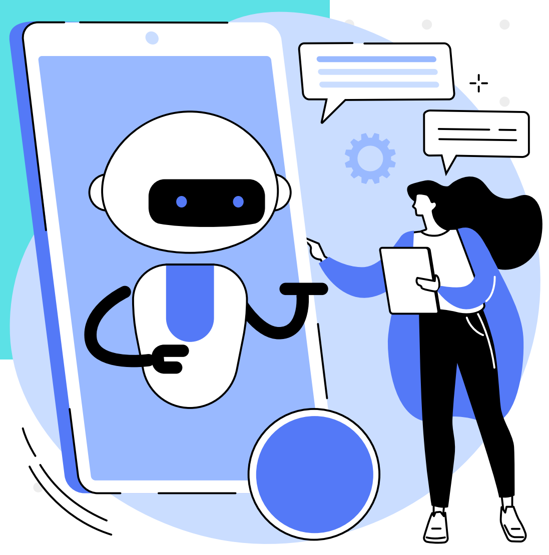
Test Your Sentences

Experiment with different types of text to observe how tokenization, POS tagging, NER, and sentiment analysis perform.

Here are two example sentences: one with positive sentiment and one with negative sentiment:

- "The new AI-powered app dramatically improved my productivity at work."
- "The latest software update caused frequent crashes and frustrated many users."

Share your pipeline's results in the comments below. Let's see how it performed on your test sentences!



8. FULL CODE and MORE!

```
import nltk

nltk.download('punkt')  # Download the tokenizer data
from nltk.tokenize import word_tokenize

#text = "Hello world. How are you?"
text = "Apple was founded by Steve Jobs and Steve Wozniak in California in 1976."

tokens = word_tokenize(text)
print(tokens)
nltk.download('averaged_perceptron_tagger_eng')
# Apply POS tagging
tagged_tokens = nltk.pos_tag(tokens)
print(tagged_tokens)

import spacy

# Load the pretrained model for NER
nlp = spacy.load("en_core_web_sm")

# Process the text with NER
doc = nlp(text)

# Extract entities
for ent in doc.ents:
    print(ent.text, ent.label_)

from transformers import pipeline

# Initialize sentiment analysis pipeline
sentiment_analyzer = pipeline('sentiment-analysis')

# Analyze the sentiment of the text
result = sentiment_analyzer(text)
print(result)
```

@Dzan Dedukic

8. FULL CODE and MORE!

Mastering these components helps you create systems that can read, understand, and interact with human language: powering everything from smart search to advanced analytics and conversational AI.

Where they're used:

- Search engines: To better understand queries and documents.
- Chatbots & virtual assistants: For understanding user intent and extracting relevant info.
- Social media monitoring: To track brand sentiment and trending topics.
- Customer support: To analyze feedback and route messages appropriately.
- News & content aggregation: To automatically tag and categorize articles.
- Compliance & information extraction: In finance, law, and healthcare, to pull out key entities and relationships from documents.

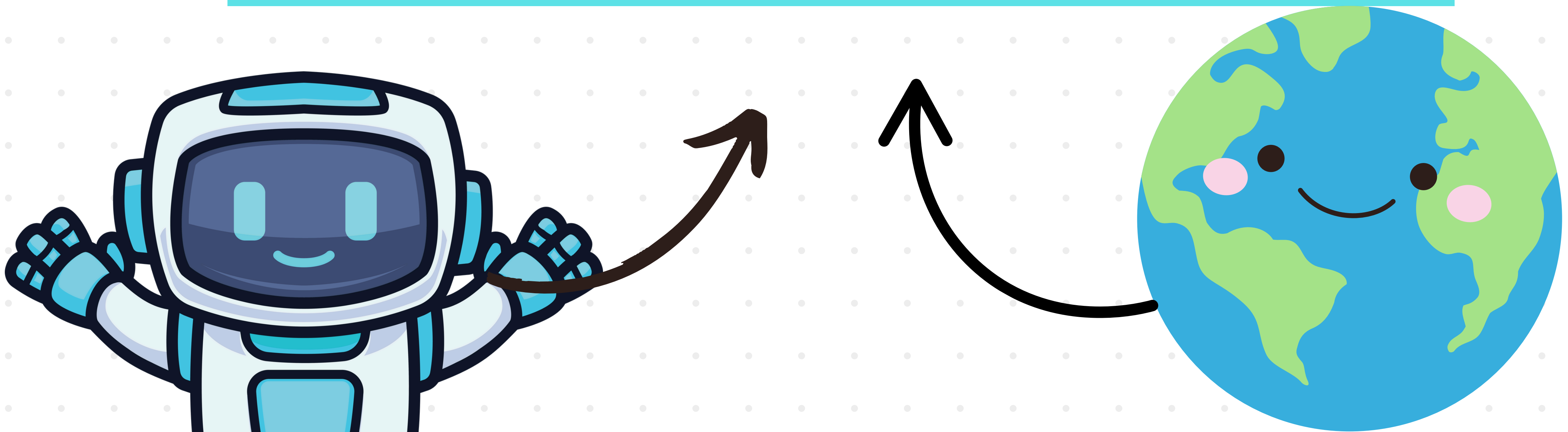
Ethical AI: Building a Transparent & Inclusive Future

🛠️ AI & ML are powerful tools—but their impact depends on how we use them.

💡 Transparency, fairness, and accountability must be at the heart of every AI-driven solution. Technology should empower, not harm, and ensure inclusivity rather than bias.

🌍 The key is in our hands. By prioritising ethical decision-making, we can build AI systems that enhance lives, foster trust, and contribute to a better world for all of humanity.

🔍 Join the AI movement that's making powerful language models accessible to everyone for more inclusive and better world.



Are you AI enthusiast as well? 🚀

Did you enjoy this carousel and find it helpful?

Follow me for more content like this, and let’s share knowledge to grow together in the world of AI! 💡✨

FOLLOW

@Dzan Dedukic