Introduction to Natural Language Processing

Linguistic Data vs. Numerical/Categorical Data

Unlike numerical and categorical data, which is highly structured with predefined meanings and organization, linguistic data presents unique challenges and advantages.

Challenges:

- Unstructured nature with no predefined organization or format
- Difficult to process and interpret programmatically
- Complex understanding required to infer context, pronouns, and word meanings
  e.g., "When the chef brought the irate diner the dish again, he told him it was still too cold."

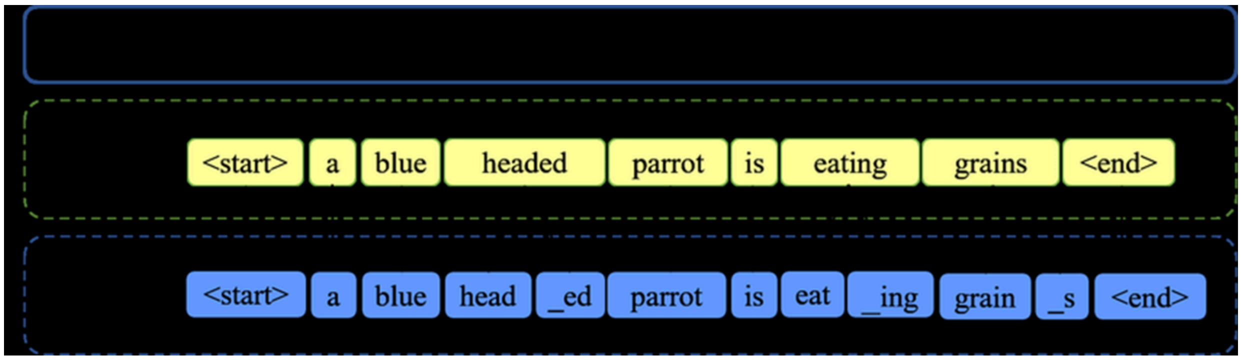
Advantages:

- Incredibly abundant due to constant human text and document generation
- Naturally contains relevant information about human lives and interests
- Rich in meaningful content and context

Tokenization

Tokenization is the fundamental first step in preparing text for machine learning models. It involves breaking continuous text into discrete chunks.

- **Words as Basic Tokens**: While humans naturally understand continuous speech, breaking text into words is already a form of tokenization
- **Subword Tokenization**: Preferred over simple word tokenization because:
  - Allows models to learn meaningful subcomponents (like suffixes)
  - Enables understanding of new words based on known components
  - Example: Breaking "climbed" into "climb" + "ed" helps the model understand past tense construction



Text Cleaning & Preprocessing

Text preprocessing can involve various operations depending on the specific needs of the model.

Common Preprocessing Steps:

- **Stop words**: removing “meaningless” words (‘a’, ‘an’, ‘the’, ‘for’, ‘but’, etc.)
- **Stemming**: removing suffixes (‘running’ → ‘run’, ‘helped’ → ‘help’)
- **Lemmatization**: converting related words to their base form (‘better’ → ‘good’)
- **Lower-casing**: removing case information

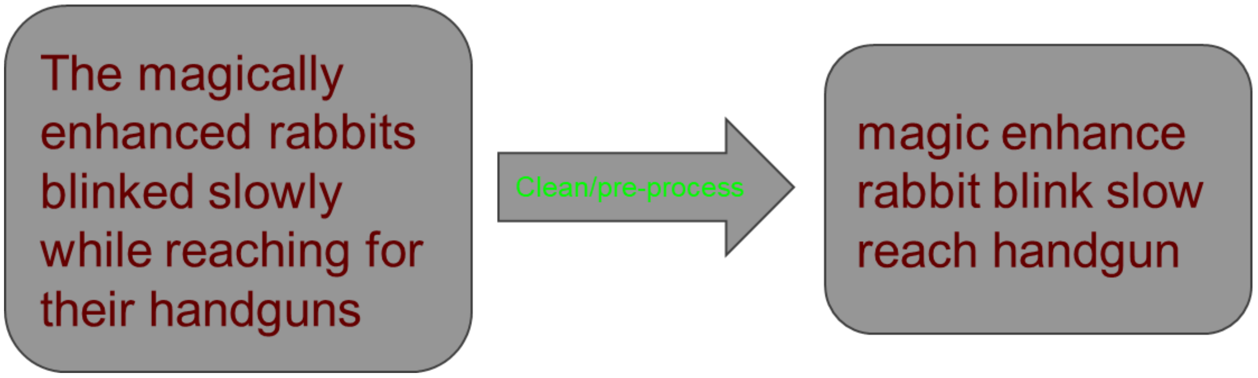
When to Use Heavy Preprocessing:

- For specific, targeted tasks
- When working with smaller datasets
- When simplifying the model's learning requirements

When to Avoid Heavy Preprocessing:

- For general-purpose models (like large language models)
- When working with vast amounts of data
- When maintaining nuanced meaning is important

Note: The level of preprocessing should be balanced against the intended use case. While heavy preprocessing can simplify the model's task, it may also remove valuable information that could be necessary for more complex understanding.

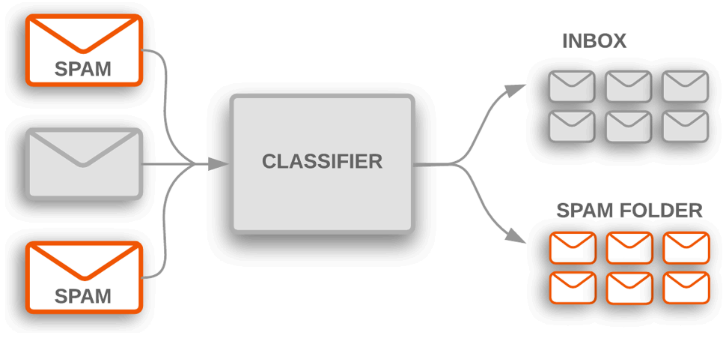

The magically enhanced rabbits blinked slowly while reaching for their handguns
Clean/pre-process
magic enhance rabbit blink slow reach handgun

Tasks in Natural Language Processing

Text Classification

Text Classification is a process where pieces of text are categorized into predefined categories based on their content.

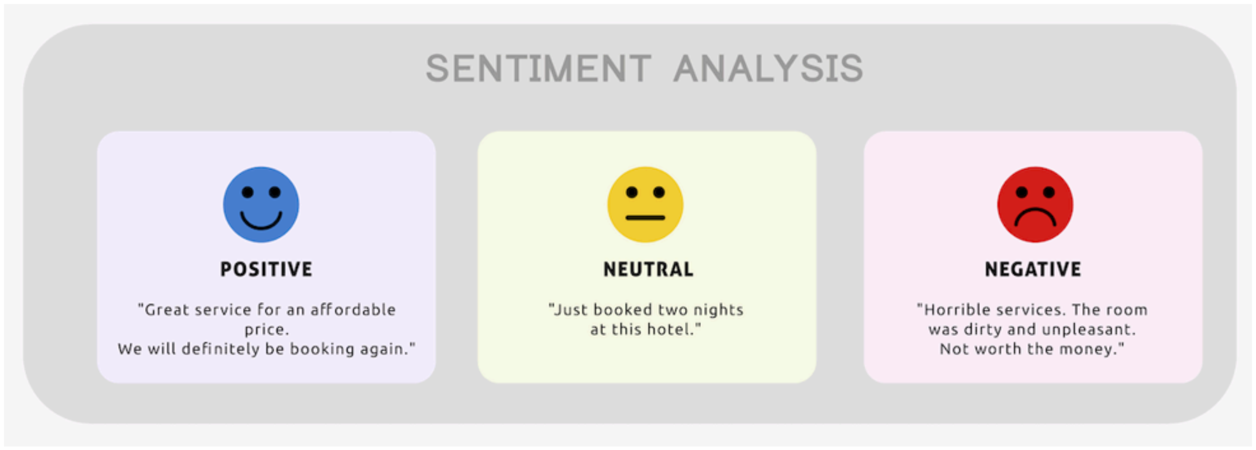
- Common examples: Spam detection in emails, news categorization, etc.
- Model selection depends on data size and task specificity:
  - Simple tasks/small data: Naive Bayes, Support Vector Machines
  - Complex tasks/large data: Convolutional Neural Networks, Long Short-Term Memory Networks



Sentiment Analysis

Sentiment Analysis is a specialized form of text classification focusing on emotional tone.

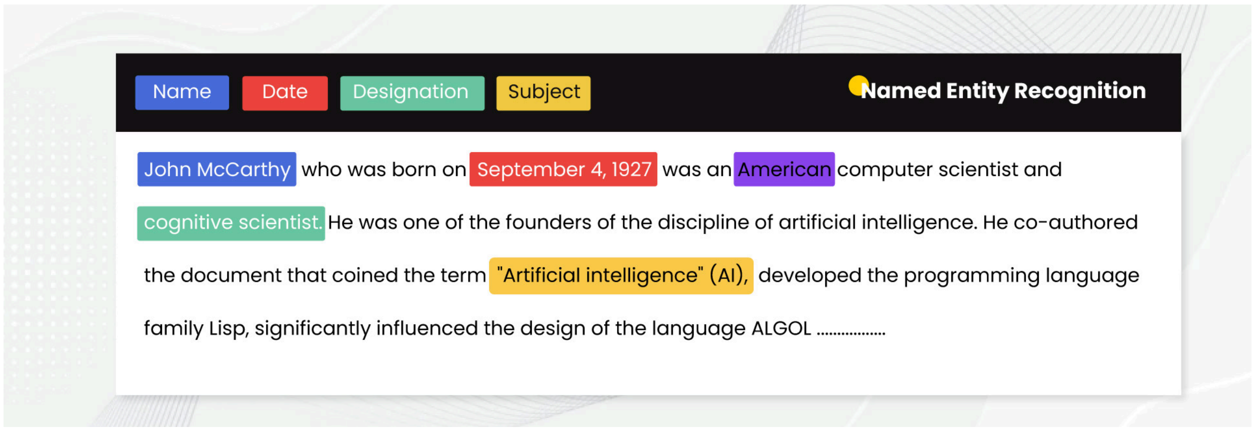
- Typically categorizes text as positive, negative, or neutral
- Common applications:
  - Brand monitoring on social media
  - Public opinion analysis for elections
  - Policy sentiment tracking
- Implementation approaches:
  - Simple: Lexicon-based models (counting positive/negative words)
  - Complex: Sophisticated AI models for nuanced analysis


SENTIMENT ANALYSIS
POSITIVE
"Great service for an affordable price.
We will definitely be booking again."
NEUTRAL
"Just booked two nights at this hotel."
NEGATIVE
"Horrible services. The room was dirty and unpleasant.
Not worth the money."

Named Entity Recognition (NER)

Named Entity Recognition (NER) is an NLP task that identifies and classifies named entities (such as names, organizations, locations, dates, and monetary values) within unstructured text.

- Applications:
  - Metadata extraction
  - Question-answering systems
  - Information retrieval
- Uses pre-trained models capable of recognizing new entities
- Example capability: Can identify organization names even if not in training data
- May occasionally misclassify ambiguous names or terms



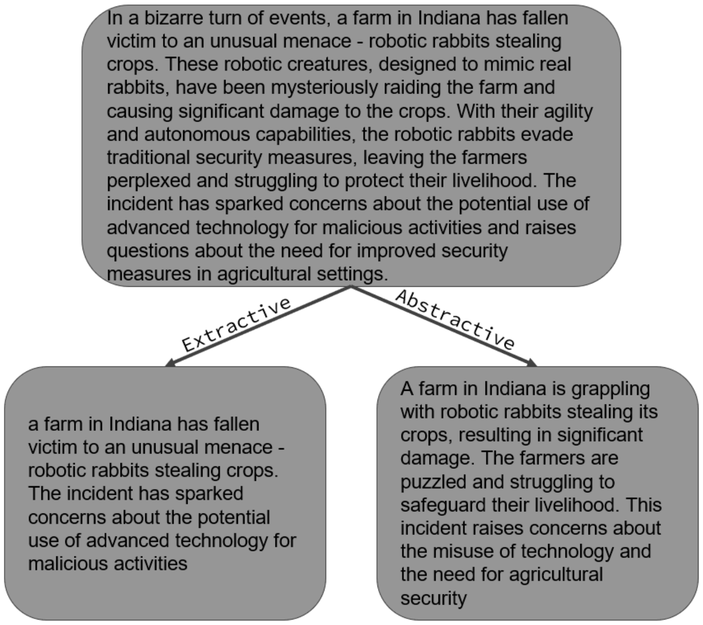
John McCarthy who was born on September 4, 1927 was an American computer scientist and cognitive scientist. He was one of the founders of the discipline of artificial intelligence. He co-authored the document that coined the term "Artificial intelligence" (AI), developed the programming language family Lisp, significantly influenced the design of the language ALGOL ....

Text Summarization

Text summarization is the automated process of creating a shorter, coherent version of a longer text while preserving its key information and meaning.

There are two main approaches:

1. **Extractive Summarization**
   - Selects and combines existing sentences from the original text
   - Simpler to implement
   - Requires less computational power
   - Works well with limited data
2. **Abstractive Summarization**
   - Generates new text for the summary
   - Requires more sophisticated models
   - Needs ability to generate coherent text
   - More similar to human-style summarization

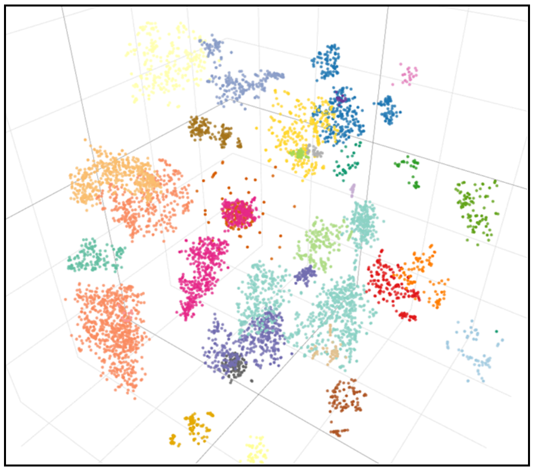


In a bizarre turn of events, a farm in Indiana has fallen victim to an unusual menace - robotic rabbits stealing crops. These robotic creatures, designed to mimic real rabbits, have been mysteriously raiding the farm and causing significant damage to the crops. With their agility and autonomous capabilities, the robotic rabbits evade traditional security measures, leaving the farmers perplexed and struggling to protect their livelihood. The incident has sparked concerns about the potential use of advanced technology for malicious activities and raises questions about the need for improved security measures in agricultural settings.

a farm in Indiana has fallen victim to an unusual menace - robotic rabbits stealing crops. The incident has sparked concerns about the potential use of advanced technology for malicious activities

A farm in Indiana is grappling with robotic rabbits stealing its crops, resulting in significant damage. The farmers are puzzled and struggling to safeguard their livelihood. This incident raises concerns about the misuse of technology and the need for agricultural security

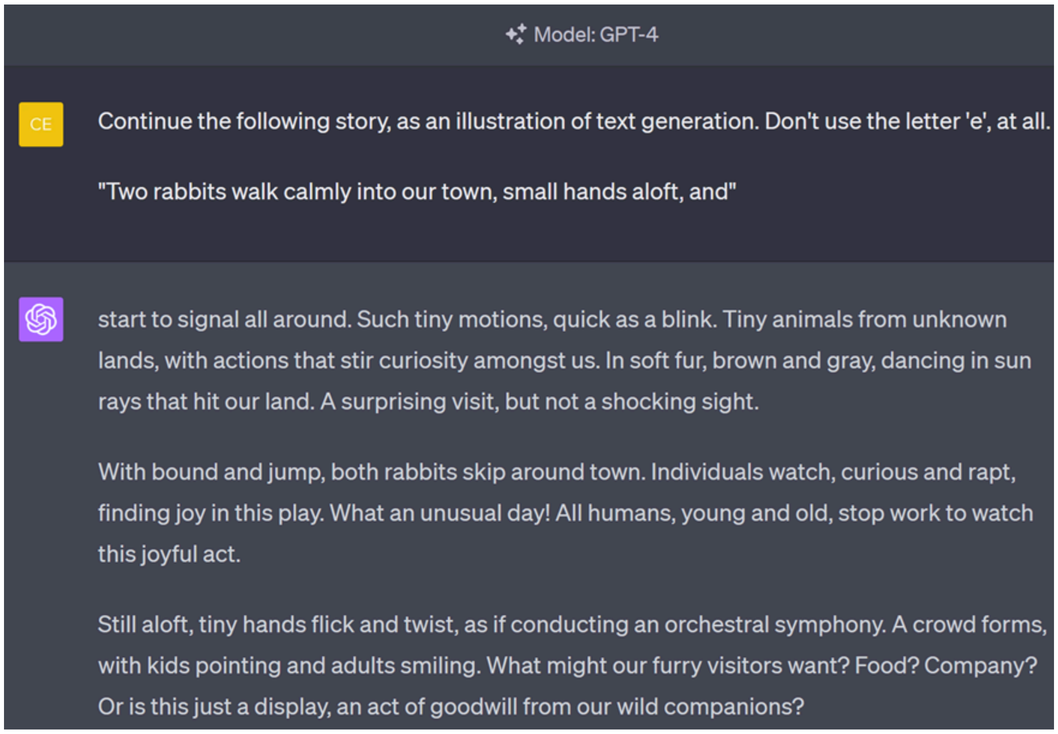
Topic Modeling



- Useful for analyzing large text corpora
- Clusters documents into related topics
- Applications:
  - Analyzing large datasets (e.g., millions of tweets)
  - Identifying discussion themes in forums
  - Providing overview of large text collections
- Helps in understanding content without reading every document

Text Generation

- Current state-of-the-art technology in NLP
- Evolution of technology:
  - Past: Recurrent Neural Networks, particularly LSTM networks
  - Present (post-2017): Transformer-based deep neural networks
- Major platforms:
  - GPT models
  - Google's PaLM
  - Claude
  - Llama
- Capabilities continuously expanding with new use cases being discovered
- Shows remarkable proficiency in generating human-like text



Continue the following story, as an illustration of text generation. Don't use the letter 'e', at all.

"Two rabbits walk calmly into our town, small hands aloft, and"

start to signal all around. Such tiny motions, quick as a blink. Tiny animals from unknown lands, with actions that stir curiosity amongst us. In soft fur, brown and gray, dancing in sun rays that hit our land. A surprising visit, but not a shocking sight.

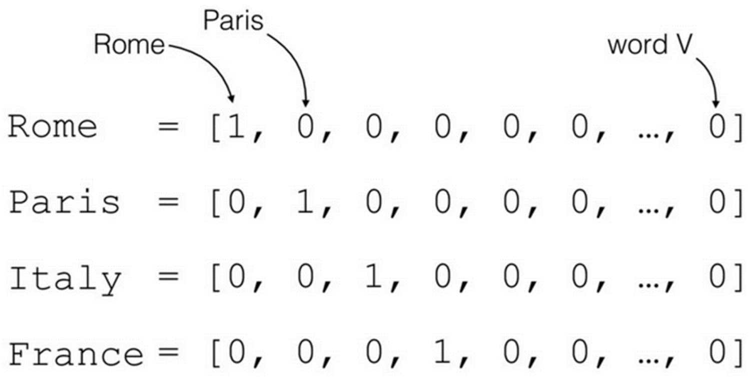
With bound and jump, both rabbits skip around town. Individuals watch, curious and rapt, finding joy in this play. What an unusual day! All humans, young and old, stop work to watch this joyful act.

Still aloft, tiny hands flick and twist, as if conducting an orchestral symphony. A crowd forms, with kids pointing and adults smiling. What might our furry visitors want? Food? Company? Or is this just a display, an act of goodwill from our wild companions?

NLP Embeddings and Models Summary

Bag of Words: Token Vectors

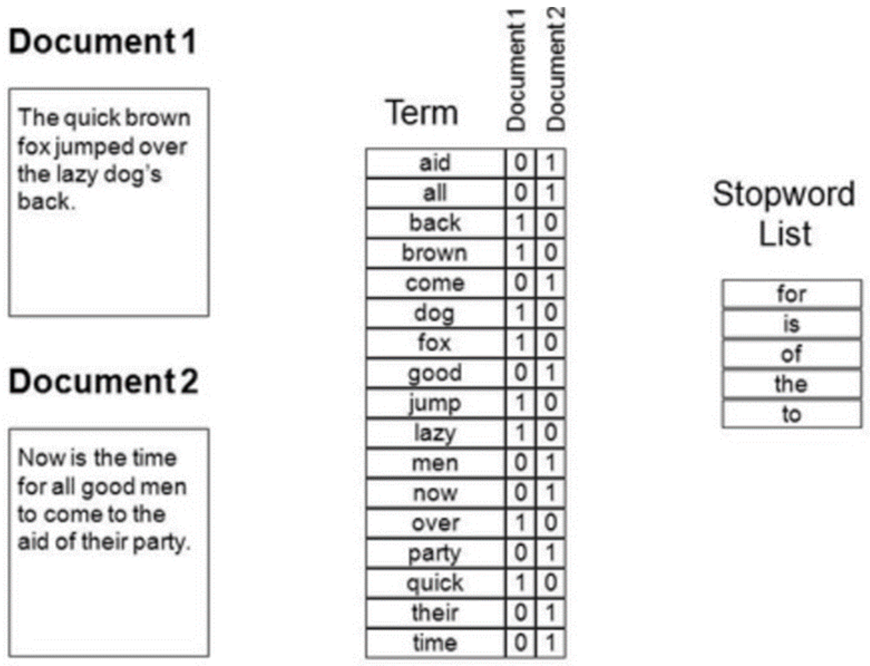
- Text must be converted to numbers for ML models to process
- Each word in vocabulary is assigned a specific position in a vector
- Vector length equals vocabulary size (e.g., 10,000 words = 10,000-length vector)
- Each position represents count of specific word occurrences



```
Rome   = [1, 0, 0, 0, 0, 0, …, 0]
Paris  = [0, 1, 0, 0, 0, 0, …, 0]
Italy  = [0, 0, 1, 0, 0, 0, …, 0]
France = [0, 0, 0, 1, 0, 0, …, 0]
```

Bag of Words: Document Vectors

- Documents represented as vectors of word counts
- Vector positions correspond to vocabulary words (often in alphabetical order)
- Example given: corpus with 22 unique words, 5 stop words removed, resulting in 17-word vocabulary
- Each document becomes vector of counts for each vocabulary word



Document 1

The quick brown fox jumped over the lazy dog's back.

Document 2

Now is the time for all good men to come to the aid of their party.

Term	Document 1	Document 2
aid	0	1
all	0	1
back	1	0
brown	1	0
come	0	1
dog	1	0
fox	1	0
good	0	1
jump	1	0
lazy	1	0
men	0	1
now	0	1
over	1	0
party	0	1
quick	1	0
their	0	1
time	0	1

Stopword List
for
is
of
the
to

Bag of Words: Pros and Cons

Pros:

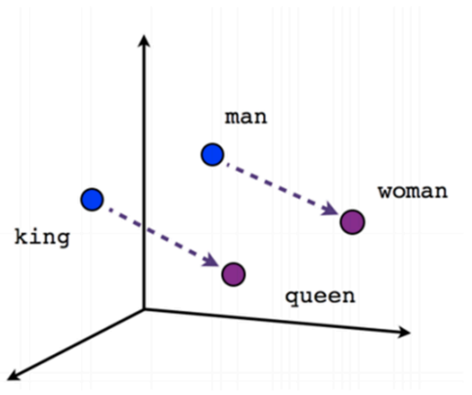
- Easy to implement
- Human-readable when vocabulary mapping is known
- Simple to get numerical representation
- Often effective for specific use cases or small datasets

Cons:

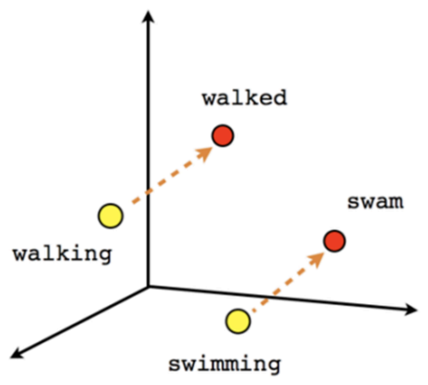
- Completely loses word order (e.g., "man bites dog" = "dog bites man")
- Very long feature vectors (vocabulary size determines length)
- Uses only non-negative integers, missing nuance of continuous values
- Sensitive to preprocessing (adding words requires rebuilding all vectors)
- Sparse vectors (mostly zeros)
- No semantic relationships between words

Word Embeddings

- Improvement over bag of words
- Captures semantic information and relationships
- Uses smaller number of features
- Represents words in continuous space rather than discrete counts



Male-Female



Verb tense

Word2Vec: Motivation and Idea

- Introduced in 2013
- Based on principle "you shall know a word by the company it keeps" - John Rupert Firth, 1957
- Words are similar if they appear near similar words
- Creates embeddings that capture semantic relationships
- Similar words (e.g., "bagel" and "latte") get similar numerical representations

Word	**Context Words**	**Word Vector**
Latte	Coffee, shop, drink, ordered, delicious, hot, cup	<0.5, 0.7, -1.2, 1.6>
Bagel	Coffee, shop, breakfast, ordered, delicious, toasted,	<0.4, 0.7, -1.1, 1.0>
Bicycle	Ride, commute, road, mountain, tire, helmet	<-0.1, 0.6, 1.2, -0.5>

Word2Vec: Projection Matrix

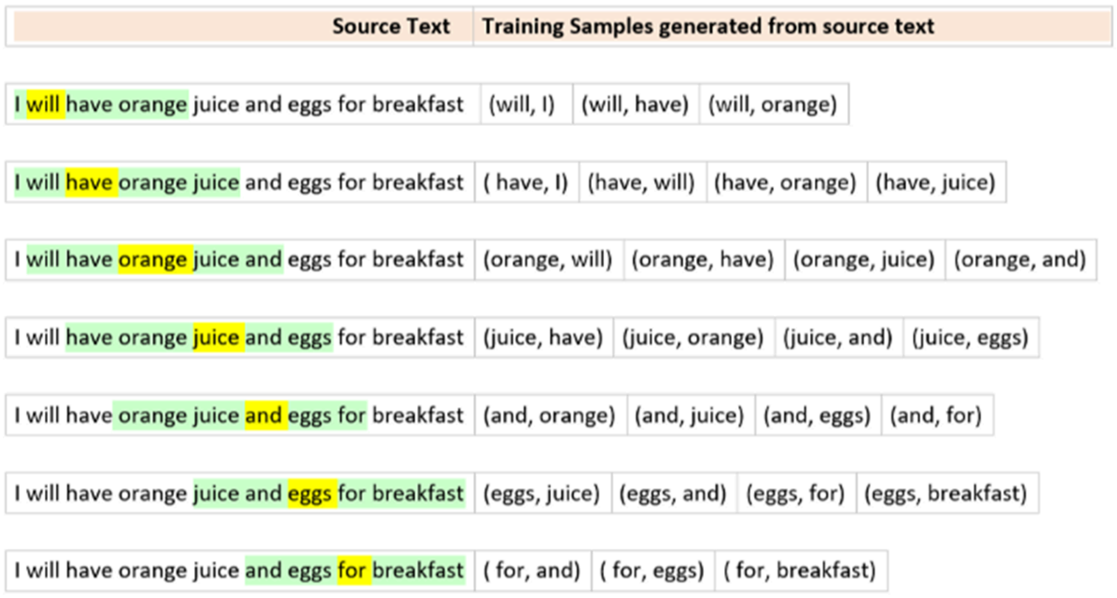
- Transforms bag of words representation through matrix multiplication
- Uses matrix W1 to project into embedding space
- Matrix learned through training process
- Converts sparse, discrete representations into dense, continuous ones

$$\mathbf{v}_{\mathrm{w2v}} = W1\,\mathbf{v}_{\mathrm{bow}}$$

Word2Vec: CBOW (Continuous Bag of Words)

- Training task: predict missing word from surrounding context
- Uses window of surrounding words (e.g., 2-3 words on each side)
- Starts with random W1 matrix
- Improves through backpropagation
- Single sentence generates multiple training examples
- Window size is configurable based on available resources

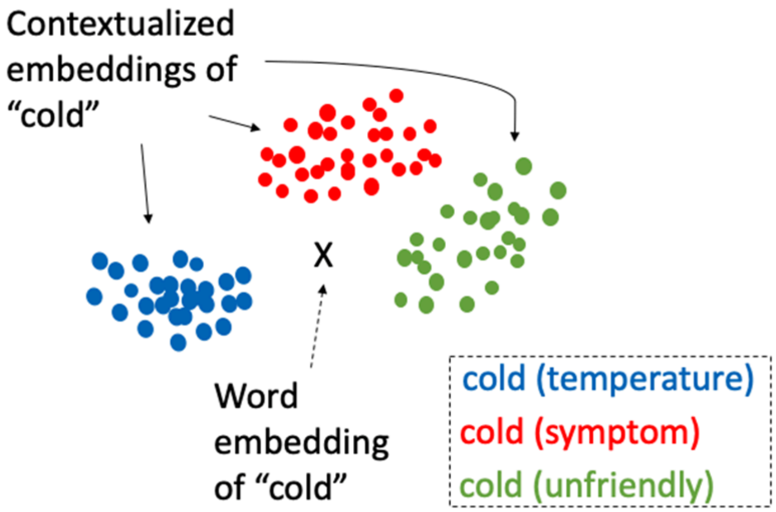
This morning, ***I ordered a*** _____ ***at the coffee*** shop. It was delicious!



Source Text	Training Samples generated from source text			
I will have orange juice and eggs for breakfast	(will, I)	(will, have)	(will, orange)	
I will have orange juice and eggs for breakfast	(have, I)	(have, will)	(have, orange)	(have, juice)
I will have orange juice and eggs for breakfast	(orange, will)	(orange, have)	(orange, juice)	(orange, and)
I will have orange juice and eggs for breakfast	(juice, have)	(juice, orange)	(juice, and)	(juice, eggs)
I will have orange juice and eggs for breakfast	(and, orange)	(and, juice)	(and, eggs)	(and, for)
I will have orange juice and eggs for breakfast	(eggs, juice)	(eggs, and)	(eggs, for)	(eggs, breakfast)
I will have orange juice and eggs for breakfast	(for, and)	(for, eggs)	(for, breakfast)	

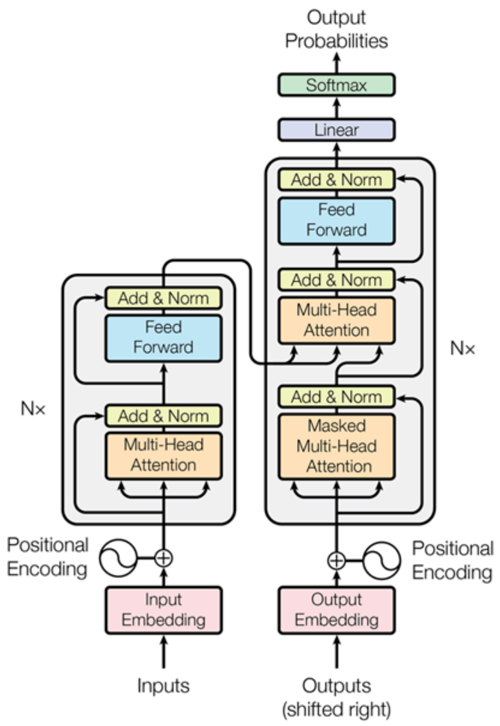
Contextual Embeddings

- Addresses limitation of static word embeddings
- Words can have different meanings based on context
- Example: "cold" in "has a cold" vs. "coffee is cold"
- Enables dynamic representation based on usage



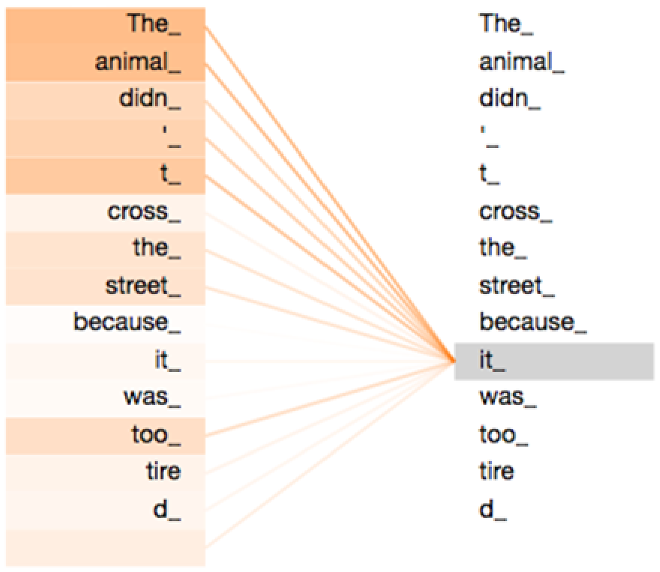
Transformers

- Introduced in 2017
- Revolutionary in NLP and expanding to computer vision
- Uses self-attention mechanism
- Considers entire context when creating embeddings
- Dominant architecture in modern NLP



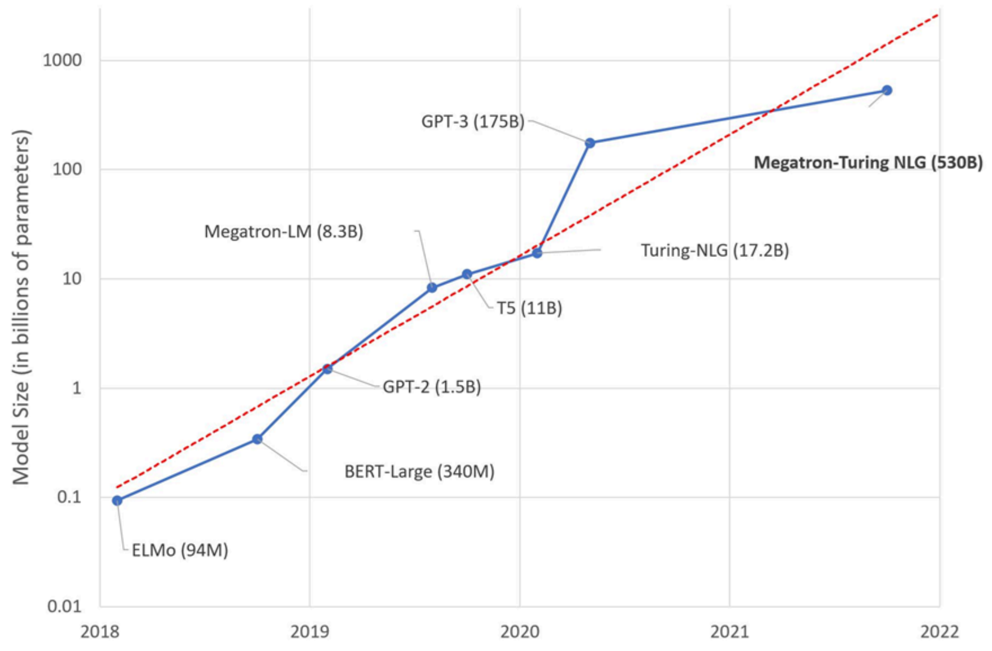
BERT

- Transformer-based model, "Bidirectional Encoder Representations from Transformers"
- Uses attention to focus on relevant context
- Creates context-sensitive embeddings
- Example: correctly identifies references for pronouns based on context
- Different from static embeddings as same word gets different representations in different contexts



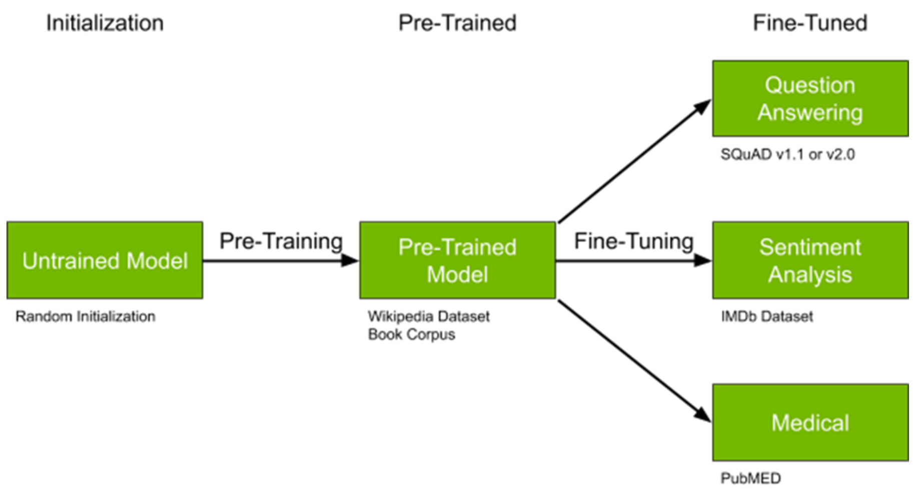
Model Complexity

- Transformer models are resource-intensive
- Require very large amounts of training data
- Better performance with larger models
- Limited to large tech companies due to resource requirements



Fine-Tuning

- Pre-train large model on massive dataset (e.g., internet text)
- Fine-tune pre-trained model on smaller, specific dataset
- Enables use of powerful models for specific tasks
- Example: training classifier for hospital incident reports using pre-trained model


Initialization
Pre-Trained
Fine-Tuned
Untrained Model
Random Initialization
Pre-Training
Pre-Trained Model
Wikipedia Dataset
Book Corpus
Fine-Tuning
Question Answering
SQuAD v1.1 or v2.0
Sentiment Analysis
IMDb Dataset
Medical
PubMED