Hate Speech Detection in Turkish using BERT-based Models

Işık Topçu

CSS Seminar Series
Represented by:
Center for Computational Social Sciences, Koç University
MA-CSSL, Koç University

March 4, 2024

Who am I?

- Master's student at Koç University in Computational Social Sciences.

Research Endeavors:

- Blame attributions on social media about economic downturns.
- Anti-immigrant and refugee discourse on social media.
- Researcher for the Politus Analytics.

Areas of Interest:

- Unsupervised topic modelling.
- BERT-based models.
- Geospatial Mapping.

Today's Schedule

1. **Hate speech definition and annotation schemes.**
2. **Case Study:** Overview of the Pittsburgh Project: Insights from the CDT dataset and its relevance to the Turkish context.
3. **Introduction to the basics of BERT and its importance in NLP.**
   - How does BERT work?
   - Overview of the life-cycle of BERT, from pre-training to fine-tuning.
4. **A hands-on session** where participants engage in training and using BERT for hate speech detection.
5. **Q&A**

Communication & Questions

- Experienced in unsupervised topic modeling techniques such as BERTopic, LDA, and GSDMM etc.
- Have worked on the Politus project, focusing on training and predicting data using transformer models like emotion, ideology, stance detection, topics, and geospatial mapping.
- Open to questions and can assist with topic modeling issues.

Contact Information:

- Email: `itopcu21@ku.edu.tr`
- GitHub: `www.github.com/isiktopcu`
- Slido: `www.slido.com` — code: 5089031

What is Hate Speech Exactly?

- **Importance of Definitions:** Defining terms correctly in annotation tasks is crucial to avoid incorrect annotations and false results.
- **Garbage In, Garbage Out:** Accurate definitions are as important as the models themselves to prevent the "garbage in, garbage out" scenario, where wrong data input leads to incorrect outputs.

What is Hate Speech Exactly?

Definition 1

"Hate speech involves expressions of hate or hostility against individuals or groups, based on race, religion, ethnicity, gender, or other attributes."
Reference: "Hate Speech, Subject Agency and Performativity of Bodies" by Ha-June Jeong.

Definition 2

"It ranges from derogatory language to inciting violence."
Reference: "Hate Speech Analysis using Supervised Machine Learning Techniques" by M. Pyingkodi et al.

What is Hate Speech Exactly? (Cont'd)

United Nations Strategy and Plan of Action on Hate Speech

Hate speech is defined as any kind of communication in speech, writing, or behavior, that attacks or uses pejorative or discriminatory language with reference to a person or a group on the basis of who they are, in other words, based on their religion, ethnicity, nationality, race, color, descent, gender, or other identity factor.

Words That Wound

"Words That Wound: Critical Race Theory, Assaultive Speech, And The First Amendment" by Matsuda et al. (1993): Hate speech is identified as speech that has the potential to inflict emotional harm or incite violence against members of historically oppressed groups. It is often targeted at characteristics such as race, gender, ethnicity, nationality, religion, sexual orientation, and the like.

Impact of Hate Speech in the Digital Era

"In the digital era, its spread on social media is a major concern as it is quite contagious."

Reference: "Hate Speech in Social Media and Its Effects on the LGBT Community: A Review of the Current Research" by Oana Ștefăniță and Diana-Maria Buf.

Impact of Hate Speech on Political Figures

More importantly, in our case study, hate speech can hurt political figures emotionally, making them feel isolated and damaged or worse: putting their candidacy at risk for the elections.

What are other types of online violence?

Other types of online violence, beyond hate speech which are out of NLP's radar, include:

- **Cyberbullying:** Repeated harassment or intimidation of an individual, often through social media or messaging platforms.
- **Cyberstalking:** Persistent tracking, monitoring, or harassment of an individual through digital means.
- **Doxing:** Publishing private or identifying information about an individual online, typically with malicious intent.
- **Trolling:** Deliberately provocative or offensive posts intended to upset or elicit an angry response from others.
- **Identity Theft:** Stealing personal information to impersonate someone online, often for financial gain or other malicious purposes.

Online Violence against Female Candidates Project Overview

Project: Joint `project` with Pittsburgh University Ford Institute for Human Security and the Politus team.

Team Members:

- Fırat Durusan
- Melih Can Yardı
- Myself

Guided by Prof. Yörük, Prof. Hürriyetoglu, Professor Steven, and Müge Finkel.

Current Phase:

- Engaged in the detailed annotation process.

Project Significance:

- A hands-on case study focusing on hate speech detection.
- Exploring the nuances of hate speech within the Turkish Twitter landscape.

Case Study Overview

- Objective: To develop deep learning models for detecting hate speech against political candidates during Turkey's 2023 elections.
- Data: Analysis of over 1.1 million tweets by approximately 176,000 users.
- Focus: Temporality, spatial distribution, and thematic content of hate speech.
- Impact: Relation of hate speech to candidates' electability and identity characteristics.

CDT Data

- The project began with the CDT dataset, which includes comprehensive data on hate speech against the American candidates.
- The dataset comes from the Center for Democracy and Technology (CDT).
- It is detailed in defining types and categories of hate speech, but it is in English.
- Ideal for studying hate speech in the American context.
- The dataset can be acquired through request.
- Volume: 100K

CDT Data Classification Categories

- **Categories for classification:**
  - **Sentiment** towards the candidate (Negative, Neutral, Positive).
  - **Narrative** focus of the tweet (Identity, Ideology, Policy, Character, Electability).
  - **Identity characteristics** discussed (Disability, Gender, Race, Religion, Sexual Orientation, Socio-economic Status, Other).
  - **Type of abuse** expressed (Demeans, Direct threat, Indirect threat, Doxing, Offensive language, Sexism, Racism, Homophobia, Ethnic slurs, Promotes violence, Sexual Assault, Sexual Content, Vandalizing, Other harassment).
  - **Presence of disinformation or misinformation.**
- Coders use the codebook as a guide for consistent and accurate classification.

Performance with CDT Data

Performance Issues of Models Using CDT Data:

- **Linguistic Differences:** Models fine-tuned on English data struggle with the complexities of Turkish language structure due to significant linguistic disparities in syntax, grammar, and semantics.
- **Annotation for Machine Learning:** The data was not specifically annotated for machine learning, impacting effective learning.
- **Vocabulary and Tokenization:** The unique vocabulary, idioms, and expressions native to Turkish are not covered by English training data. Tokenization rules differ significantly between the languages, complicating text processing.

Advantages of a Turkish-Specific Model

Why a Turkish Model Outperforms Multilingual Models:

- **Specialized Vocabulary:** The BERT Turkish Base model has a comprehensive vocabulary inclusive of a wide array of Turkish words, idioms, and expressions, enabling better understanding and processing of Turkish texts.
- **Optimized for Linguistic Nuances:** Turkish-specific models are better at capturing the unique syntactic, grammatical, and semantic nuances of Turkish, such as its agglutinative nature.

Turkish BERT based models

For more Turkish base models, please visit the following GitHub repository:

`https://github.com/stefan-it/turkish-bert`

Annotation Pipeline

- Approach: Creation of a gold standard corpus through human annotation using a balanced sample from the output of the ASELSAN-Bilkent model.
- Pipeline Modules: Narrative, Offensive Speech, Hate Speech.
- Training Data:
  - Gold standard corpus annotated by experts.
  - Enhanced using pre-existing models for initial predictions.
- Aim: To discern the direction and targets of hate speech more accurately.

Findings and Conclusions

- Evident rise in hate speech volume prior to the elections.
- Disproportionate targeting of female candidates, especially from opposition parties.
- Documentation of regional disparities in hate speech prevalence.

Word Embeddings

Let's take a journey through the history of word embeddings from the inception of text representation with TF-IDF to the latest advancements with BERT, we'll explore how natural language processing has evolved to create more sophisticated and nuanced models for understanding language.

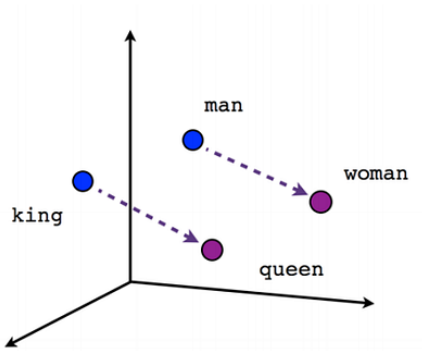
TF-IDF

TF-IDF (Term Frequency-Inverse Document Frequency)

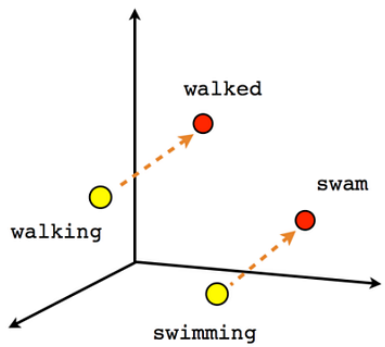
- **Concept:** A statistical measure used to evaluate the importance of a word in a document relative to a collection of documents (corpus).
- **Term Frequency (TF):** Counts the number of times a word appears in a document, signaling its significance.
- **Inverse Document Frequency (IDF):** Decreases the weight of terms that appear more frequently across the corpus, adding significance to rarer terms.
- **Equation:** $\text{TF-IDF} = \text{TF}(t, d) \times \text{IDF}(t)$
  - Where $\text{TF}(t, d)$ is the term frequency of term t in document d,
  - And $\text{IDF}(t)$ is the inverse document frequency of term t.
- **Purpose:** To reflect how important a word is to a document in a collection, useful for search and information retrieval.

Word2Vec

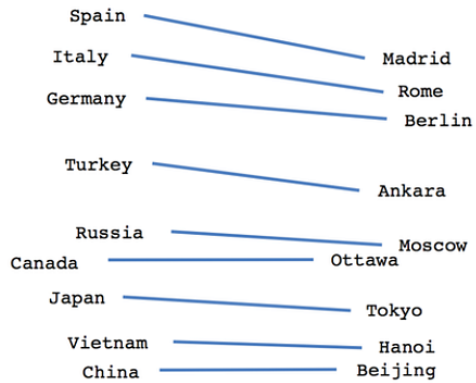
- Word2Vec represents words in a high-dimensional vector space, capturing semantic relationships.
- Similar words are positioned closely within the vector space, enabling nuanced language understanding.
- While it doesn't directly indicate the mood of text, it effectively groups words by context and usage.



Male-Female



Verb tense



Country-Capital

Issue with Word2Vec: Fixed Embeddings

- Word2Vec assigns a fixed vector to each word, regardless of context.
- This means the word *fair* would have the same representation in both
  He didn't receive fair treatment.
  and
  Fun fair in New York City this summer.
  despite the different meanings.
- Such fixed embeddings can't capture the polysemous nature of words, where a word has multiple meanings based on context.

BERT: Contextual Language Understanding

- BERT, short for Bidirectional Encoder Representations from Transformers, revolutionizes context understanding in NLP.
- Unlike Word2Vec, BERT considers the full context of a word by analyzing the words that come before and after it in a sentence.
- This bidirectional context awareness allows BERT to differentiate word meanings based on use—understanding "bark" in "dog's bark" differently from "tree bark."

BERT Contextualized Embeddings: Examples 1

- BERT generates unique embeddings for words based on their context.
- It distinguishes between multiple meanings of words like "fair" and "unbiased".

He didn't receive fair treatment.

$$\begin{bmatrix} 0.9 \\ 1.0 \\ 0.3 \end{bmatrix}$$

Tom deserves unbiased judgement.

$$\begin{bmatrix} 0.8 \\ 0.9 \\ 0.2 \end{bmatrix}$$

These embeddings reflect the contextual nuance of words in different sentences.

BERT's use of Word Embeddings

Definition

A **Word Embedding** is a feature vector representation of a word, encapsulating semantic meaning in a high-dimensional space.

- Each word is represented by a dense vector of real numbers.
- Semantic similarities are reflected in geometric closeness within this space.

Lookup Process

1. Convert word to a unique ID using a dictionary.
2. Use the ID to retrieve the word's embedding from the lookup table.

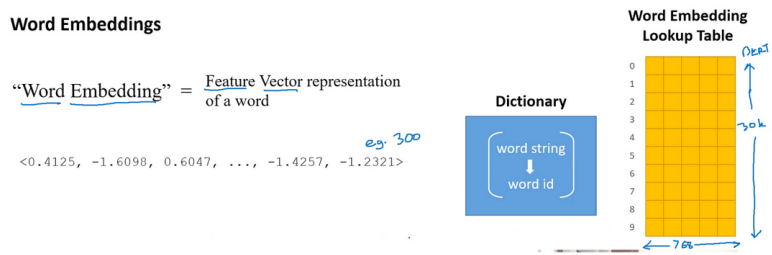


Figure: Word Embedding and Lookup Table

BERT's Use of Word Embeddings

Word Embeddings form the input layer of BERT, encoding the meaning of words into vectors.

- BERT starts with word embeddings similar to the ones shown.
- It then adjusts these embeddings through its layers to account for context.
- The resulting embeddings are *contextualized*—each word vector is influenced by the other words in the sentence.

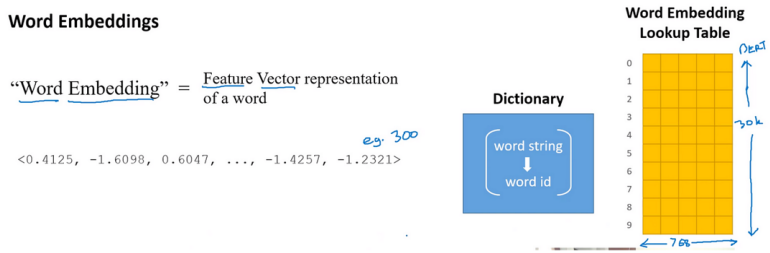


Figure: Static Word Embeddings and Lookup Table

The Process

BERT enhances static embeddings by:

1. Using the **Transformer** architecture for deep bidirectional context.
2. Applying **attention** mechanisms to assess the influence of each word on the others.
3. Producing embeddings that represent both the word's meaning and its context within a sentence.

BERT's Contextual Awareness

Conclusion

BERT's ability to generate context-specific embeddings confirms that it is contextually aware. By considering the entire context of a sentence, BERT can:

- Differentiate between the various meanings of a single word.
- Capture the subtleties of language that depend heavily on context.
- Provide more accurate representations for natural language understanding tasks.

This innovative approach marks a significant advancement over previous models, offering deeper insights into the nuances of natural language.

Why Use BERT?

- BERT understands the context of words **dynamically**, providing unique embeddings based on the surrounding text.
- It processes text **bidirectionally**, capturing nuanced language patterns and relationships between sentences.
- Pre-trained on a **vast corpus**, BERT grasps a broad spectrum of language nuances, significantly enhancing NLP tasks.

BERT in Google Search

- BERT has been integrated into Google Search to better understand the context of search queries.
- This model can predict and complete queries by understanding the relationship between words in a sentence.
- BERT's ability to process words in relation to all the other words in a sentence, rather than one-by-one in order, allows it to interpret the intent behind search queries more effectively.
- Example: When you start typing "hello" in Google, BERT helps predict what you might be looking for, such as "hello in Turkish" or "hello neighbor," by considering the context of your search.

BERT's Vocabulary - Part 1

BERT's vocabulary is a comprehensive set of tokens, including:

- **Words:** Complete words common in the training corpus.
- **Subwords:** Parts of words to handle longer words or rare terms, e.g., "unhappiness" $\rightarrow$ "un", "happi", "ness".
- **Characters:** Individual characters covering words or subwords not explicitly in the vocabulary.

These elements enable BERT to efficiently process a vast vocabulary without needing to know every possible word.

BERT's Vocabulary - Part 2

- **Special Tokens:** BERT uses special tokens for sentence structures and specific tasks:
  - **[CLS]**: For the beginning of the text, representing the whole sequence.
  - **[SEP]**: To separate sentences, especially in tasks with two sentences.
  - **[PAD]**: To fill in blank spaces for batch sequence uniformity.
  - **[MASK]**: Used in the Masked Language Model (MLM) for predicting masked tokens.

*Tokens like [CLS] and [MASK] are critical for tasks like classification and MLM in BERT.

Words, positions, and word IDs

Position	0	1	2	3	4
Word	[CLS]	the	art	of	science
Word ID	101	1996	2396	1997	2671
Vector	*Vector representations (not shown)*				

String	**Word ID**
[CLS]	101
[SEP]	102
the	1996
of	1997
art	2396
science	2671

Understanding the Input Representation in BERT

- **Position**: Index of words in a sentence, crucial for the model to recognize word order.
- **Word**: Actual tokens from the sentence. Special tokens like [CLS] and [SEP] are used to indicate the start and separation of sentences.
- **Word ID**: Unique identifiers for each token, obtained by looking up the token in BERT's vocabulary.
- **Vector**: The numerical representation of tokens used by the model to understand and process language. For each token, a vector captures various aspects of its meaning.
- The **vocabulary** table maps tokens to their corresponding IDs. BERT uses this mapping to convert text data into a numerical form it can work with.

Criticisms of BERT Models

- **Size and Complexity:** BERT models are criticized for their large size, making them capable of encoding biases and requiring substantial computational resources.
- **Interpretability:** The complexity of BERT makes it challenging to interpret how decisions are made, leading to concerns over transparency.
- **Resource Intensity:** Deploying BERT models involves significant financial and computational costs, limiting accessibility for smaller entities or projects.
- **Documentation and Usability:** There is criticism over the documentation of BERT models, which can hinder their effective utilization and adaptation in diverse applications.

These points highlight the need for balanced consideration in the deployment and development of large language models like BERT.

Fundamental Concepts

Perceptron

A basic classifier that takes an input vector and returns 1 or 0.

Layer

One step in a neural network, which may be stacked in the case of deep neural networks.

Transformer

A class of NLP model that builds representations of words in parallel.

Attention

A mechanism for automatically deciding which nearby words should influence a word's representation.

Understanding BERT's Input Processing

- **Tokenization:** Input text is broken down into word pieces, with individual words possibly split into multiple pieces.
- **Special Tokens Added:** Tokens such as [CLS] and [SEP] are added to signify the beginning and separation of sentences.
- **Word Embeddings:** Tokens are transformed into word embeddings, capturing semantic information learned during BERT's pretraining.
- **Segmentation Vectors:** Embeddings are combined with vectors representing sentence segmentation, aiding in distinguishing different segments of the input.

This initial processing sets the stage for deep contextual understanding by BERT.

BERT's Encoding and Prediction Mechanism

- **Position Vectors:** Added to embeddings to incorporate the notion of word order within the model's understanding.
- **Encoder Blocks:** A series of encoders process the input, with each encoder containing:
  1. An attention mechanism, highlighting important words.
  2. Feed-forward networks, further processing the representations.
- **Normalization and Addition:** Operations within encoders ensure smooth integration of information.
- **Final Layers:** Convert word vectors into probability distributions over potential labels, facilitating tasks like hate speech detection.

Each step, from attention to prediction, is designed to refine the understanding and representation of the input text.

BERT Processing Overview

- **Input Text Processing:**
  1. Input text is tokenized into words or subwords.
  2. Special tokens are added to the input.
- **Word Embeddings:**
  - Tokens are converted into word embeddings.
  - Embeddings incorporate segmentation and position vectors.

BERT Encoder and Prediction

- **Encoders and Attention:**
  1. A stack of encoders processes the embeddings.
  2. Attention mechanisms identify relevant words.
- **Feed Forward and Classification:**
  1. Feed-forward networks process the encoded information.
  2. Additional layers convert word vectors to probability distributions for classification.
- Goal: Understand complex operations simplified through normalization and attention.

Pretraining BERT

- **Learning the Numbers:**
  1. Pretraining involves adjusting parameters on a large dataset, like Wikipedia.
  2. Goal is to generalize across a broad range of tasks.

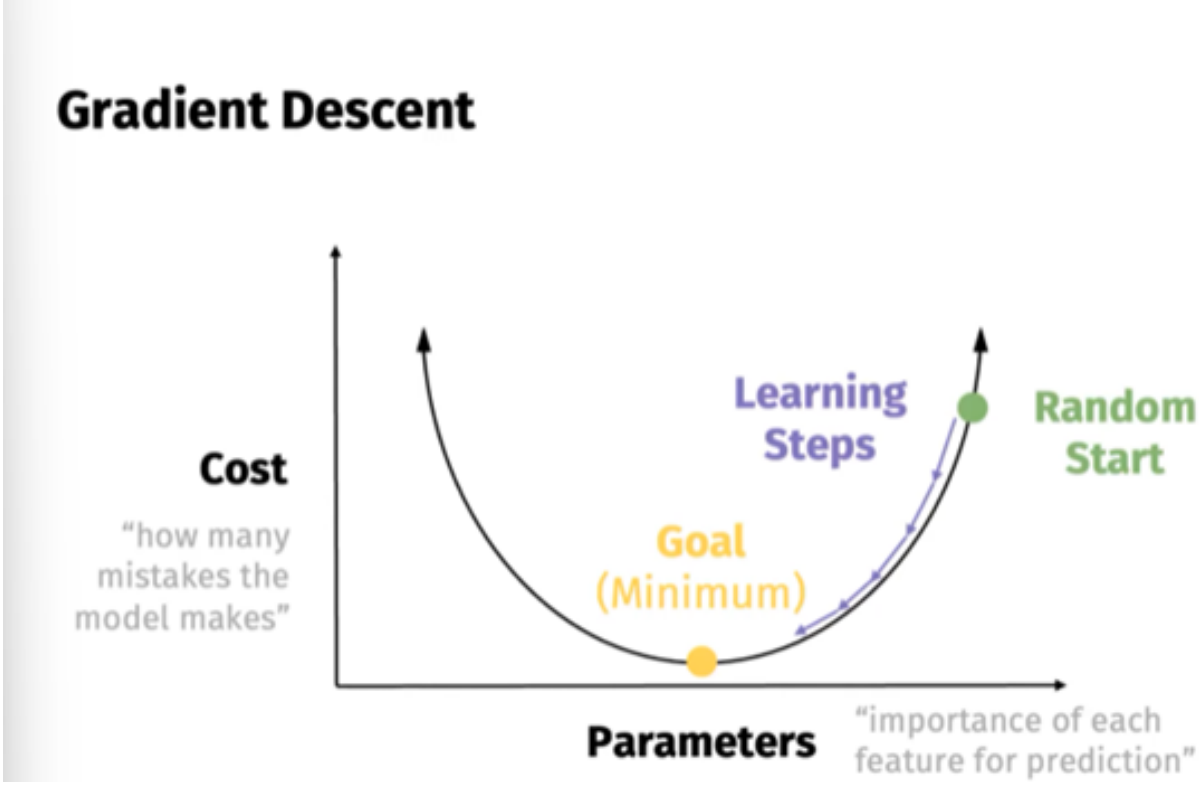
Fine-Tuning BERT

- **Adapting to Specific Tasks:**
  1. Fine-tuning adjusts the pretrained model to specific datasets or tasks.
  2. Minor tweaks to the model parameters enhance performance on target data.
- **Task-Specific Layers:**
  - Additional layers can be added for specific tasks like classification or named entity recognition.

Learning in BERT: Gradient Descent

BERT learns through **gradient descent**, a fundamental optimization technique in machine learning.

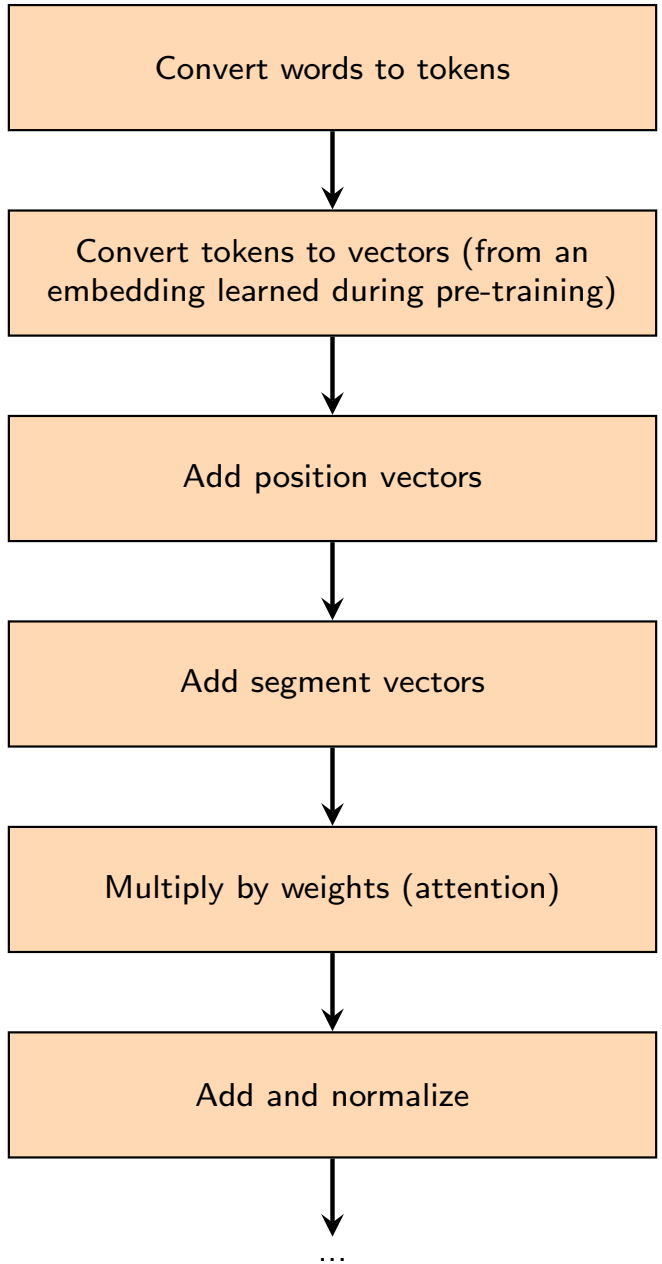
- **Gradient Descent:** Iterative process to minimize a cost function, aiming to reduce classification errors and improve model performance.
- Initially, parameters are set randomly, then updated step by step towards minimizing **the cost function**.



Fine-tuning Hyperparameters (HuggingFace)

```
training_args = TrainingArguments(
    num_train_epochs=3, # how many times to pass over the full dataset
    per_device_train_batch_size=16, # how many examples to process at a time
    per_device_eval_batch_size=20, # how many examples to process at a time
    learning_rate=5e-5, # the size of the learning steps
    warmup_steps=10, # number of steps for a linear warmup from 0 to learning_rate
    weight_decay=0.01, # multiply all weights by this term to prevent weights from getting too big
    output_dir='./results',
    logging_dir='./logs',
    logging_steps=10,
    evaluation_strategy='steps'
)
```

BERT Encoding Process Visualization



BERT Encoding Process - Part 1

The BERT model processes input text through multiple layers, transforming the original tokens into rich contextual representations. Here's the first part of the process:

1. **Token Conversion**: Words are first converted into tokens, which are then transformed into vectors using pre-trained embeddings.
2. **Segment and Position Encoding**: Each token vector is enhanced with segment and position vectors to encode the order and sentence membership information.
3. **Attention Mechanism**: In the encoder, the attention mechanism weighs the influence of different tokens on each other.

BERT Encoding Process - Part 2

Continuing from the first part, here's how BERT further processes the input:

4. **Feed Forward Layers**: After attention calculation, the representations are passed through layers of perceptrons (feed forward neural networks) for further processing.
5. **Addition and Normalization**: At several points in the process, the results are normalized and residual connections are added to help the model train more effectively.
6. **Repetition Across Encoders**: This entire process is repeated across multiple encoder layers, allowing the model to develop increasingly sophisticated representations of input text.

Case Study: Hate Speech Detection Over Multiple Contexts

- Google Colab provides free access to GPU resources to speed up computation-intensive tasks.
- To enable GPU in your Colab notebook:
  1. Go to "Runtime" in the menu.
  2. Select "Change runtime type".
  3. Choose "GPU" from the hardware accelerator dropdown.
- In this case study, we utilize Python for developing machine learning models capable of detecting hate speech across various contexts.
- Access the code and datasets for this study on GitHub:
  - Repository: `https://github.com/itopcu/bert-seminar`
  - Use the command `git clone https://github.com/itopcu/bert-seminar` to clone the repository in Google Colab.
- Ensure to install the necessary libraries and dependencies before running the code.

Dataset Information

The dataset used in this case study is an amalgamation of three human-annotated hate speech datasets. Each dataset provides labeled instances of text, where the label '1' indicates hate speech and '0' indicates non-hate speech.

- **Israeli-Palestinian Conflict Dataset** (`israel`):
  - Source: `https://github.com/isiktopcu/bert-seminar/main/Train/isr_pal_train.csv`
  - Labels and Count: Non-hate Speech (0) - 1360, Hate Speech (1) - 880
- **Refugee Crisis Dataset** (`refugee`):
  - Source: `https://github.com/isiktopcu/bert-seminar/main/Train/refugee_train.csv`
  - Labels and Count: Non-hate Speech (0) - 4477, Hate Speech (1) - 1447
- **Turkish-Greek Political Tensions Dataset** (`greek`):
  - Source: `https://github.com/isiktopcu/bert-seminar/main/Train/tr_gr_train.csv`
  - Labels and Count: Non-hate Speech (0) - 555, Hate Speech (1) - 421

What Are We Going to Do

1. **Read the Data**: Load the dataset from various sources to form the working dataset.
2. **Preprocess**: Perform necessary preprocessing steps to clean and prepare the data for modeling.
3. **Split into Train and Test Samples**: Divide the data into training and testing sets to ensure model generalizability.
4. **Oversample**: Address class imbalance by oversampling the minority class in the training data.
5. **Fine-tune**: Adjust the BERT model's parameters to best fit the training data.
6. **Get Predictions**: Use the fine-tuned model to make predictions on the test set and evaluate the results.

Each step is crucial in developing a robust model for hate speech detection.