[word2vec](https://colab.research.google.com/github/tensorflow/docs/blob/master/site/en/tutorials/text/word2vec.ipynb%23scrollTo=xP00WlaMWBZC)

Me: The technical process of creating a TensorFlow model for AI-based MP

ChatGPT: Creating a TensorFlow model for an AI-based MP would involve the following technical process:

1. Data Collection: The first step in building a TensorFlow model for an AI-based MP would be to collect data from various sources. This could include parliamentary speeches, voting records, public statements, social media activity, and other relevant information.

2. Data Preprocessing: Once the data has been collected, it would need to be preprocessed and cleaned to remove any noise or irrelevant information. This would involve tasks such as data normalization, feature extraction, and data labelling.

3. Natural Language Processing (NLP): Natural language processing (NLP) would be a critical component of the MP model. The model would need to be able to understand and interpret natural language input from users, and to generate natural language responses that are relevant and coherent. This would involve tasks such as text preprocessing, tokenization, part-of-speech tagging, and named entity recognition.

4. Model Development: With the preprocessed data and NLP components in place, an AI model could be developed using TensorFlow. The model would need to be trained to recognize patterns in the data and to learn how to generate responses based on input from users. This would involve tasks such as defining the architecture of the model, specifying the hyperparameters, and using backpropagation to optimize the model's performance.

5. Testing and Evaluation: Once the MP model has been developed, it would need to be tested and evaluated to ensure that it performs as expected. This would involve testing the model with a range of input scenarios and evaluating its output against expected responses. This would also involve measuring the accuracy and other performance metrics of the model.

6. Deployment: Finally, the MP model would need to be deployed in a production environment, such as a chatbot or virtual assistant, where it could interact with users in a simulated parliamentary context. This would involve integrating the model into an existing software system or building a new application around it.

It's important to note that building an AI-powered MP would raise significant ethical and legal questions. For example, there may be concerns around the accountability and transparency of an AI-powered representative. Additionally, there may be challenges around bias and fairness, particularly if the AI model is trained on biased or incomplete data. As such, careful consideration would need to be given to the development and deployment of such a model.

Me: on the subject of data collection: what format does this need to be in for TensorFlow

ChatGPT: TensorFlow supports various input data formats, but the format required would depend on the specific type of model being developed. In the case of building an AI-based MP, the data could be in various formats, including but not limited to:

1. Text: If the data to be collected is in the form of parliamentary speeches, voting records, public statements, or social media activity, it could be in the form of text documents. These text documents could be in formats such as plain text, PDF, HTML, or XML.

2. Images: If the data to be collected is in the form of medical images or other visual data, it could be in the form of image files such as PNG, JPEG, or TIFF.

3. Audio: If the data to be collected is in the form of audio recordings, it could be in the form of audio files such as WAV or MP3.

Once the data is collected in its original format, it would typically need to be preprocessed and transformed into a format that can be used by TensorFlow. This could involve converting text data into numerical representations, resizing, and normalizing image data, or converting audio data into spectrograms or other numerical representations. Ultimately, the format required for TensorFlow would depend on the specific model architecture and requirements.

Me: I will give you an example of a text document from the UK parliament (1971-04-01): 1. Mr. Leonard asked the Secretary of State for Education and Science how many primary school classes of more than 40 children there are in the London Borough of Havering; and what steps are being taken to eliminate overcrowding in primary schools in the borough. The Under-Secretary of State for Education and Science (Mr. William van Straubenzee) In January, 1970, 150. Figures for 1971 are not readily available but should be lower. 560 additional places will be provided by two new primary schools approved for starting in the period up to March, 1973. Mr. Leonard I welcome the progress which has been made, but is the hon. Gentleman aware that there are no grounds for satisfaction about the present position in Havering, which has easily the worst record for over-crowded classes of any London borough? Will the hon. Gentleman impress on the borough that to reduce the size of over-crowded classes must take priority over cutting local rates, which the council has perversely chosen to do this year? Mr. van Straubenzee I have no doubt that the position in Havering, which I accept is unsatisfactory, is very much in the mind of the local education authority.

ChatGPT: Thank you for providing an example of a text document from the UK parliament. This text document could be used as a valuable source of information for developing an AI-based MP. The next step would be to preprocess this text data so that it can be used by TensorFlow. This could involve several steps, such as:

1. Tokenization: Breaking the text down into individual words or tokens.

2. Stopword removal: Removing common words such as "the", "and", and "of" that do not add much meaning to the text.

3. Stemming/Lemmatization: Reducing words to their root form to account for variations in word forms.

4. Vectorization: Converting the text data into numerical representations that can be fed into a machine learning model.

Once the text data has been preprocessed, it can be used to train a TensorFlow model for the specific task of predicting how a Member of Parliament would respond to similar questions. This model would likely be based on a neural network architecture that is trained using a large amount of text data, such as parliamentary speeches and voting records.

Me: Could you show me a processed version of the text

ChatGPT: Certainly! Here's an example of how the text document you provided could be preprocessed:

Original text:

Mr. Leonard

asked the Secretary of State for Education and Science how many primary school classes of more than 40 children there are in the London Borough of Havering; and what steps are being taken to eliminate overcrowding in primary schools in the borough.

The Under-Secretary of State for Education and Science (Mr. William van Straubenzee)

In January, 1970, 150. Figures for 1971 are not readily available but should be lower. 560 additional places will be provided by two new primary schools approved for starting in the period up to March, 1973.

Mr. Leonard

I welcome the progress which has been made, but is the hon. Gentleman aware that there are no grounds for satisfaction about the present position in Havering, which has easily the worst record for over-crowded classes of any London borough? Will the hon. Gentleman impress on the borough that to reduce the size of over-crowded classes must take priority over cutting local rates, which the council has perversely chosen to do this year?

Mr. van Straubenzee

I have no doubt that the position in Havering, which I accept is unsatisfactory, is very much in the mind of the local education authority.

Processed text:

mr leonard ask secretary state education science many primary school classes 40 children london borough havering steps taken eliminate overcrowding primary schools borough undersecretary state education science mr william van straubenzee january 1970 150 figures 1971 not readily available lower 560 additional places provided two new primary schools approved starting period march 1973 mr leonard welcome progress made hon gentleman aware no grounds satisfaction present position havering easily worst record overcrowded classes london borough hon gentleman impress borough reduce size overcrowded classes must take priority cutting local rates council perversely chosen year mr van straubenzee no doubt position havering accept unsatisfactory much mind local education authority

As you can see, the processed text has been converted to lowercase, tokenized, and stop words have been removed. This processed text can now be further preprocessed and used to train a TensorFlow model.

Me: So how does the preprocessed text look like

ChatGPT: Here's an example of how the preprocessed text might look like after further processing for use in a TensorFlow model:

[('mr', 'leonard'), ('ask', 'secretary'), ('state', 'education'), ('science', 'many'), ('primary', 'school'), ('classes', '40'), ('children', 'london'), ('borough', 'havering'), ('steps', 'taken'), ('eliminate', 'overcrowding'), ('primary', 'schools'),...]

This is a list of tuples where each tuple contains two words that occur together in the preprocessed text. This type of data is often used in natural language processing and can be fed into a TensorFlow model for further training and analysis.

Me: which TensorFlow model do I use for the data

ChatGPT: The choice of TensorFlow model to use for your data would depend on the specific task you are trying to accomplish. Based on the preprocessed text data you provided, some potential models that could be used include:

- Word2Vec: This model can be used to learn word embeddings, which are vector representations of words that capture their semantic meaning. Word2Vec can be used for tasks such as text classification, sentiment analysis, and similarity matching.

- Recurrent Neural Network (RNN): This model can be used for tasks such as language modelling, text generation, and sentiment analysis. RNNs are particularly well-suited for processing sequential data, such as text.

- Convolutional Neural Network (CNN): This model can be used for tasks such as text classification, sentiment analysis, and topic modelling. CNNs are particularly good at identifying patterns in local regions of input data, such as individual words or phrases.

The choice of model would depend on the specific task you want to accomplish and the characteristics of your data. It's important to note that building an effective TensorFlow model often requires significant experimentation and tuning to find the best architecture and hyperparameters for your specific use case.

Me: I then loaded VS Code created a folder called Python. I then created a file called skip-gram.ipynb

I then loaded the code from the website mentioned at the top of the document. I ran into a few errors. I continued to use ChatGPT as follows:

Me: # Load the TensorBoard notebook extension %load\_ext tensorboard

ChatGPT: This command is used in Jupyter notebooks to load the TensorBoard extension, which allows you to visualize and analyse TensorFlow models. Once the extension is loaded, you can launch the TensorBoard web application from within the notebook and use it to explore your model's performance, metrics, and graphs.

To use this command, you need to have the TensorBoard package installed in your environment. You can install it using the following command:

!pip install tensorboard

Once the package is installed, you can use the `%load\_ext tensorboard` command to load the extension and start using TensorBoard in your notebook.

Me: ModuleNotFoundError: No module named 'tqdm'

ChatGPT: The `ModuleNotFoundError: No module named 'tqdm'` error occurs when the `tqdm` module is not installed in your Python environment. The `tqdm` module is a package that provides a progress bar to track the progress of iterative tasks.

To resolve this error, you can install the `tqdm` module using `pip`. Open your terminal or command prompt and run the following command:

pip install tqdm

Once the `tqdm` module is installed, you should be able to import it in your Python script without any issues.

Me: ModuleNotFoundError: No module named 'tensorflow'

ChatGPT: The `ModuleNotFoundError: No module named 'tensorflow'` error occurs when the `tensorflow` package is not installed in your Python environment.

To resolve this error, you can install TensorFlow using `pip`. Open your terminal or command prompt and run the following command:

pip install tensorflow

Make sure you have the appropriate permissions to install packages. If you are using a virtual environment, activate it before running the above command.

Once the installation is complete, you can import TensorFlow in your Python script without any issues.

Me: ERROR: Could not install packages due to an OSError: [WinError 5] Access is denied: 'C:\\Users\\infob\\AppData\\Local\\Programs\\Python\\Python311\\Lib\\site-packages\\~umpy\\.libs\\libopenblas64\_\_v0.3.21-gcc\_10\_3\_0.dll' Consider using the `--user` option or check the permissions.

ChatGPT: The `ERROR: Could not install packages due to an OSError: [WinError 5] Access is denied` error occurs when you don't have sufficient permissions to install packages in the target directory.

You can try to resolve this error by running the command with administrator privileges. Open your terminal or command prompt as an administrator and try running the `pip install` command again:

pip install tensorflow

Alternatively, you can use the `--user` option to install the package in the current user's home directory:

pip install --user tensorflow

If the above solutions don't work, you may need to check the permissions for the target directory and make sure you have write access to it.

Me: WARNING: The scripts estimator\_ckpt\_converter.exe, import\_pb\_to\_tensorboard.exe, saved\_model\_cli.exe, tensorboard.exe, tf\_upgrade\_v2.exe, tflite\_convert.exe, toco.exe and toco\_from\_protos.exe are installed in 'C:\Users\infob\AppData\Roaming\Python\Python311\Scripts' which is not on PATH. Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.

ChatGPT: This warning message indicates that the scripts installed by TensorFlow are located in a directory that is not in your system's PATH environment variable.

You can add the directory `C:\Users\infob\AppData\Roaming\Python\Python311\Scripts` to your system's PATH environment variable so that the scripts can be easily executed from the command line.

Here are the steps to add a directory to your PATH variable in Windows:

1. Open the Start menu and search for "Environment Variables" and click on the "Edit the system environment variables" option.

2. In the System Properties window, click on the "Environment Variables" button.

3. In the Environment Variables window, locate the "Path" variable in the "System variables" section and click on "Edit".

4. Click on "New" and add the directory path `C:\Users\infob\AppData\Roaming\Python\Python311\Scripts` to the list of paths.

5. Click "OK" to close all the windows.

After completing these steps, the system will recognize the scripts installed by TensorFlow and the warning message should no longer appear.

Me: I have run this Skip-gram and negative sampling can I show you the result

ChatGPT: Sure! Please share the results of your Skip-gram and negative sampling implementation.

Me: target : 15 context : tf.Tensor([14 0 4 9 2], shape=(5,), dtype=int64) label : tf.Tensor([1 0 0 0 0], shape=(5,), dtype=int64)

ChatGPT: It looks like you have successfully implemented Skip-gram with negative sampling!

The `target` value of 15 indicates that the central word being predicted is the 15th word in the vocabulary.

The `context` values are the indices of the surrounding words within a window of context size. In this case, the context words are `[14 0 4 9 2]`.

The `label` values are the binary labels indicating whether each context word is a true context word or a negative sample. In this case, the first context word is the true context word and the remaining four are negative samples, as indicated by the value `1` for the first element and `0` for the remaining elements.

Overall, this output indicates that the model is correctly identifying the central word and generating the appropriate context words and labels for training.

Me: can you give me a more detail meaning of this

ChatGPT: Certainly!

In this example, we have a target word with an index of 15. The context words are represented by a tensor with indices [14, 0, 4, 9, 2]. These context words are the words that surround the target word in the training data. The label tensor shows whether each context word is a positive or negative example for the target word. In this case, the first context word with index 14 is a positive example (assigned a label of 1) while the other context words are negative examples (assigned a label of 0).

The Skip-gram with Negative Sampling (SGNS) algorithm is designed to train a neural network to predict the probability of a target word appearing given a context word or vice versa. The model learns to map each word in the vocabulary to a high-dimensional vector. The SGNS algorithm uses negative sampling to train the model efficiently by only updating the parameters for a small subset of negative examples, rather than all the words in the vocabulary. The output of the SGNS algorithm is a set of word embeddings, which can be used as input to other machine learning models for various NLP tasks such as text classification, sentiment analysis, and machine translation.

Me: so first what would the text classification be? then what would the sentiment be?

ChatGPT: To classify the text, we would need to define the categories or classes that the text could belong to. For example, if we were analysing political speeches, we might have categories such as "economic policy", "foreign policy", and "domestic policy". Without knowing the specific context of the text, you are referring to, it is difficult to determine appropriate categories for classification.

Once we have established the categories, we could use a text classification model in TensorFlow to classify the text. The model would take the preprocessed text as input and output a predicted category.

To determine the sentiment of the text, we would use a sentiment analysis model in TensorFlow. The model would take the preprocessed text as input and output a sentiment score, usually ranging from negative to positive.

Me: the category for the text is firstly a domestic policy and its regarding education.

ChatGPT: Thank you for providing the context of the text. Based on the content of the text you provided, it can be categorized as a question about education policy in a specific geographical region, which is the London Borough of Havering. Therefore, the category of the text can be classified as "Education Policy" or "Local Governance".

As for the sentiment, it is difficult to determine the sentiment of the text without further information. The text appears to be a question posed by Mr. Leonard regarding the steps being taken to eliminate overcrowding in primary schools in the borough. Therefore, the sentiment could be neutral, as it is a simple inquiry for information. However, Mr. Leonard's remark about the council's decision to cut local rates could be interpreted as negative, as he implies that the council is not prioritizing reducing the size of overcrowded classes.

Me: as this is only 1 very small example of a policy discussion and how we have managed to turn it into machine code with what seems to be a result of negative. Collecting all the transcripts would show positive and negative outcomes. What would we need to do for the AI MP?

ChatGPT: To create a functional AI MP, you would need to collect a large amount of data from parliamentary transcripts, including speeches, questions, and debates. This data should cover a range of topics and issues to ensure that the AI MP can effectively engage with a wide variety of constituents and understand the complexities of the political landscape.

Once the data is collected, it needs to be preprocessed to prepare it for input into a TensorFlow model. This would involve tasks such as cleaning the text, tokenization, converting the words into numerical representations, and splitting the data into training, validation, and testing sets.

Next, you would need to choose an appropriate model architecture for the AI MP. This could involve using a pre-trained language model such as BERT or GPT-3, or creating a custom model using techniques such as recurrent neural networks (RNNs) or convolutional neural networks (CNNs).

After building the model, it would need to be trained on the preprocessed data using appropriate loss functions and optimization algorithms. Once the model is trained, it would need to be fine-tuned on new data to ensure that it is able to adapt to changing political contexts and issues.

Finally, the AI MP would need to be integrated into a user interface or other platform for interaction with constituents. This would involve developing natural language processing (NLP) capabilities for the AI MP to understand and respond to user queries and feedback.