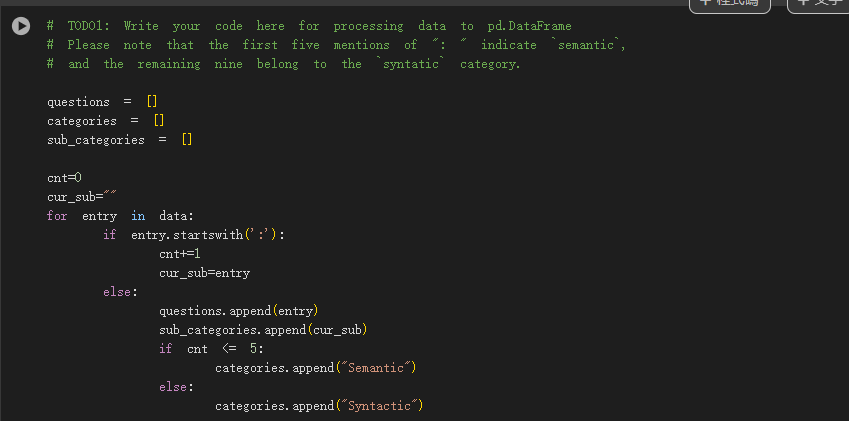
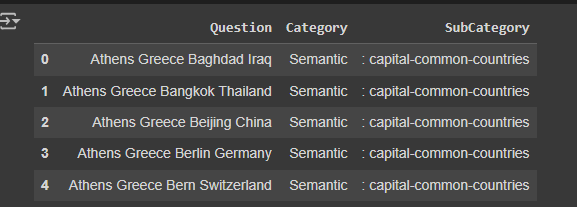
Running environment :Colab；Python version :Colab

I. Implementation results

#TODO1:Convert the analogy data to pd.DataFrame (evaluation dataset)

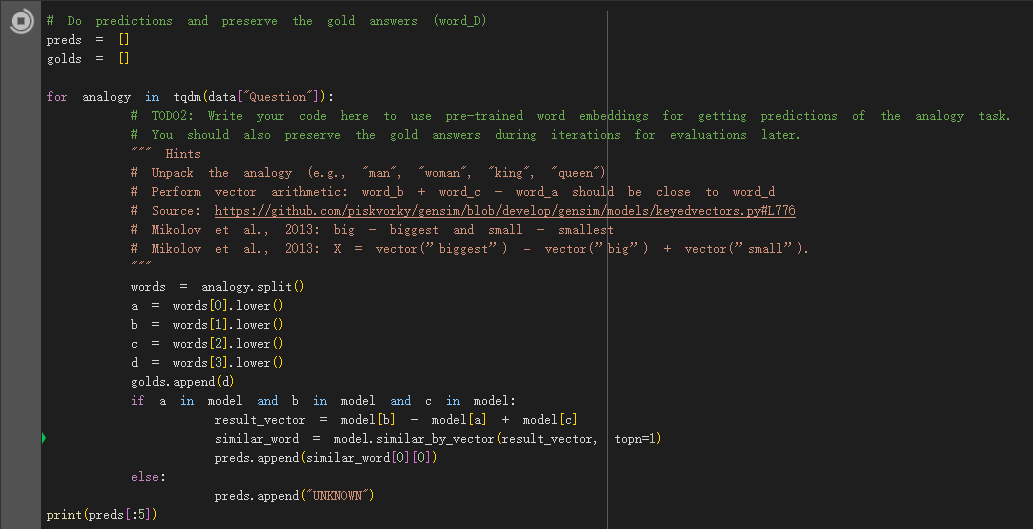
Save the three fields of each question and set the first five subcategories to semantic and the rest to syntatic.





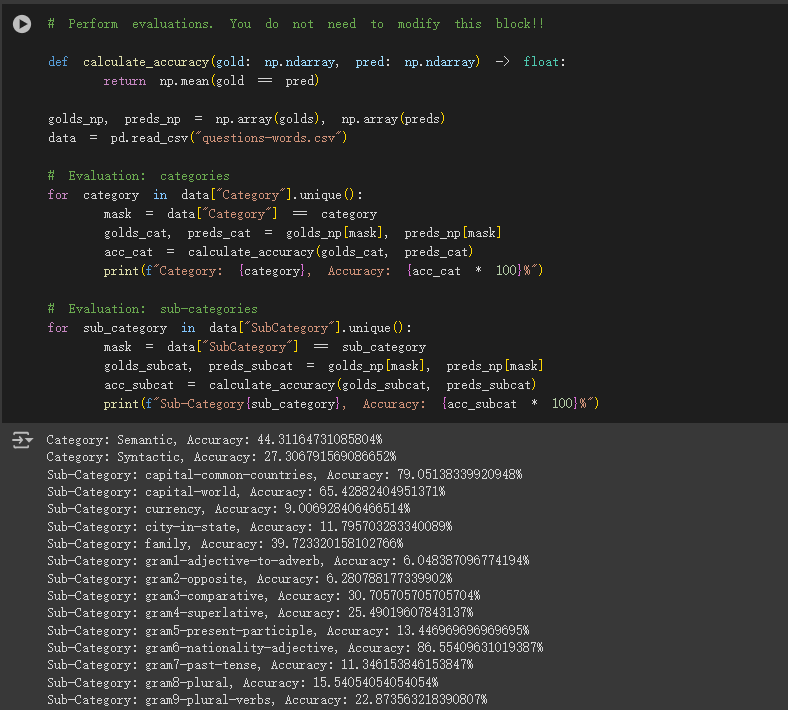
#TODO2:Get predictions of the analogy task using pre-trained word embeddings

Directly predict vocabulary by vector addition and subtraction, and store the correct solution in the gold list. Note that only lowercase words can be used as indexes.

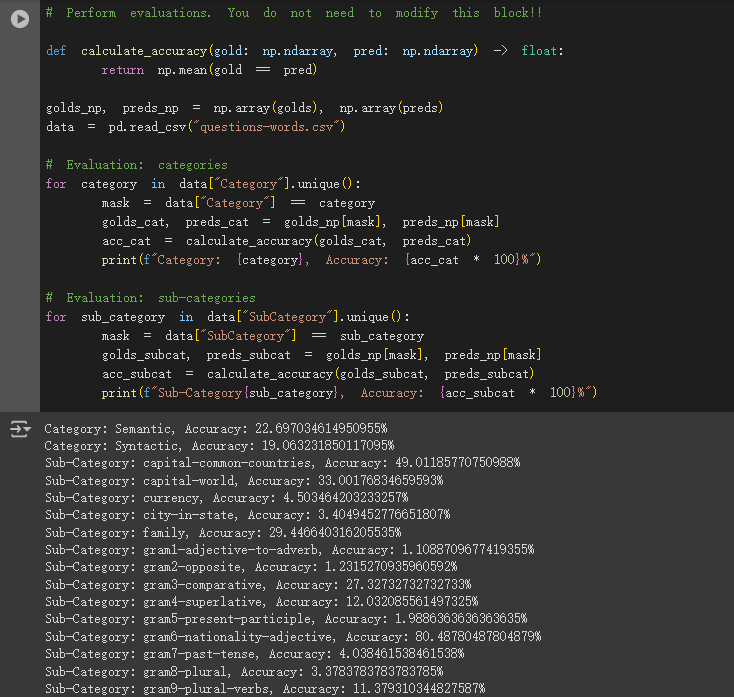


Prediction result:

glove-wiki-gigaword-100:

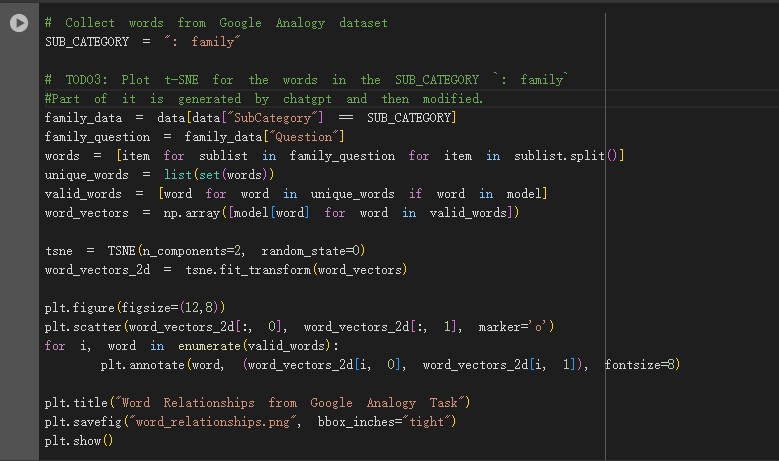


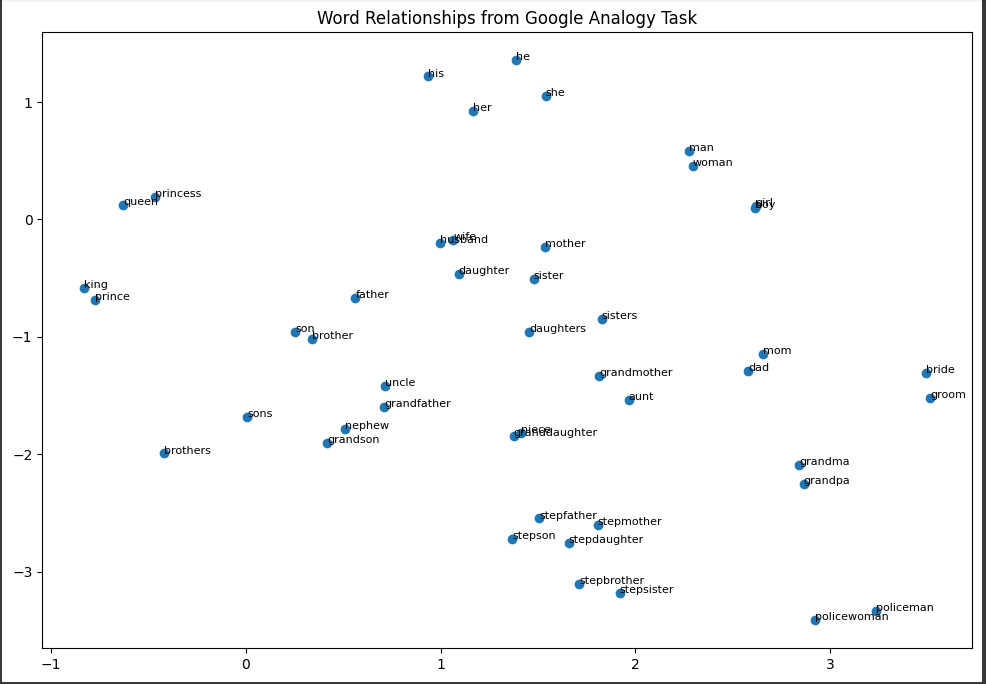
glove-wiki-gigaword-300:



#TODO3:Plot t-SNE to see word relationships in the sub-category of `family`

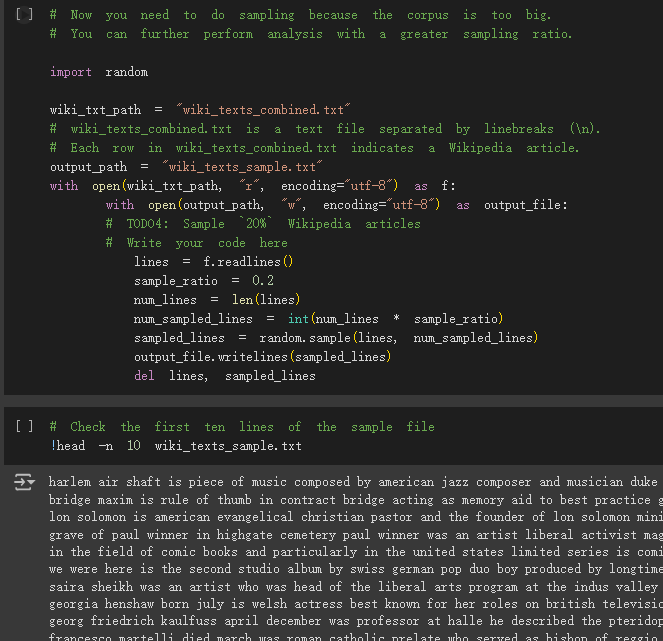
Use tsne method to reduce the dimensionality of data and visualize it.( Part of it is generated by chatgpt and then modified.)





#TODO4:Sample 20% Wikipedia articles

First read all the data, and then use the random package to randomly sample 20% of them as training data.



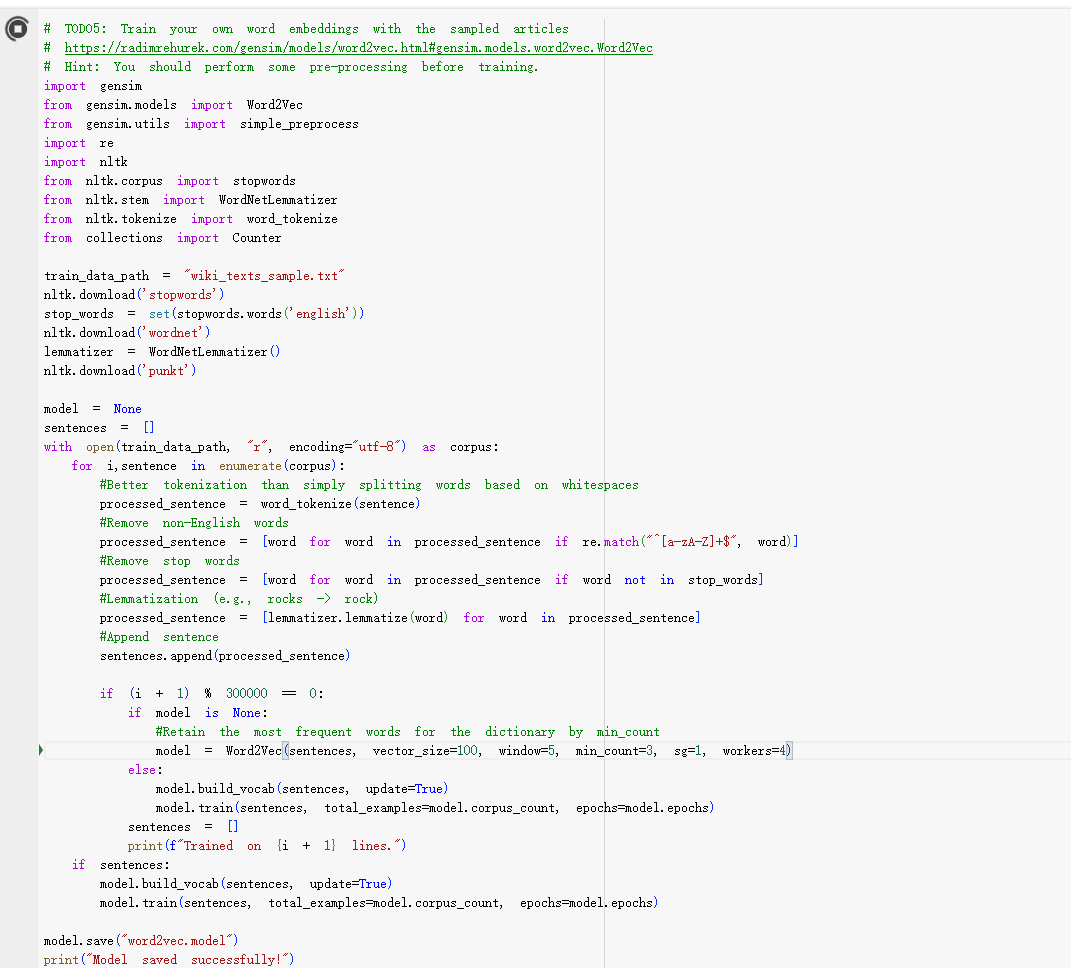
#TODO5 Train your own word embeddings with the sampled articles

#TODO6 Get predictions of the analogy task using your trained word embeddings

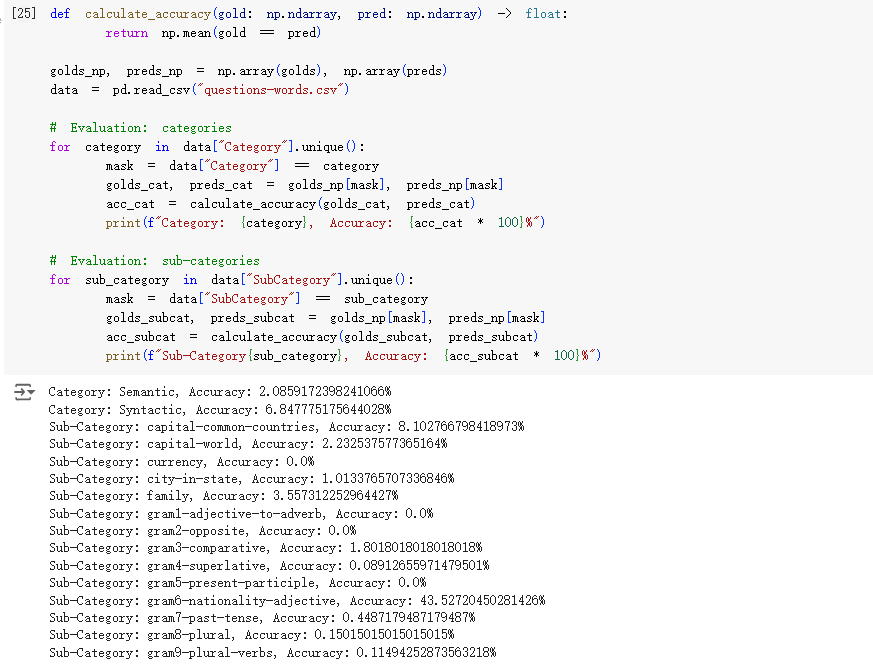
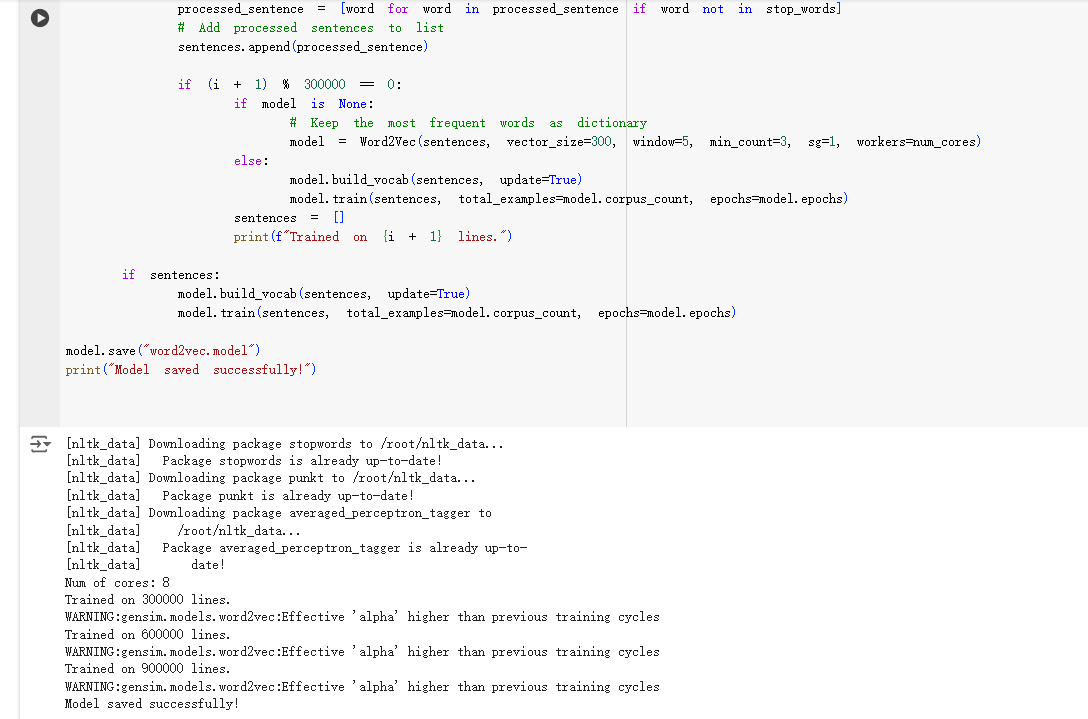
(Most of them are generated by chatgpt, and then some code and related parameters are adjusted.)

I tried various parameter combinations and here are my results:

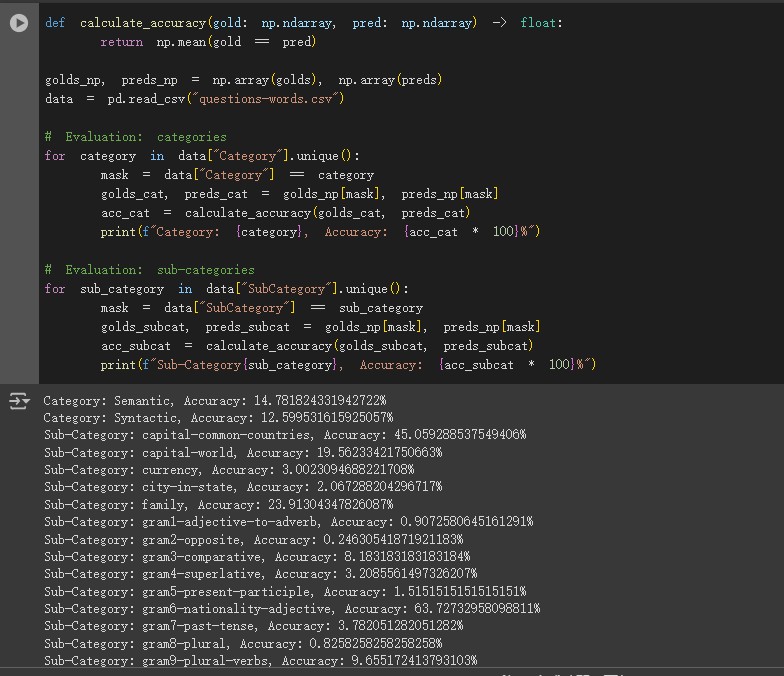
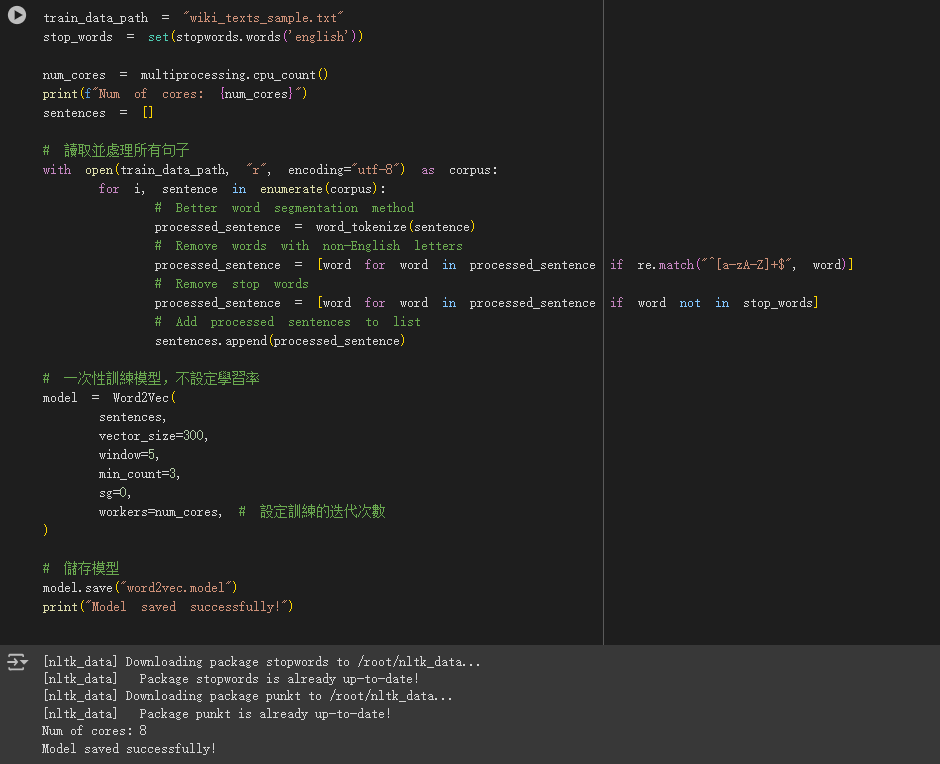
1.



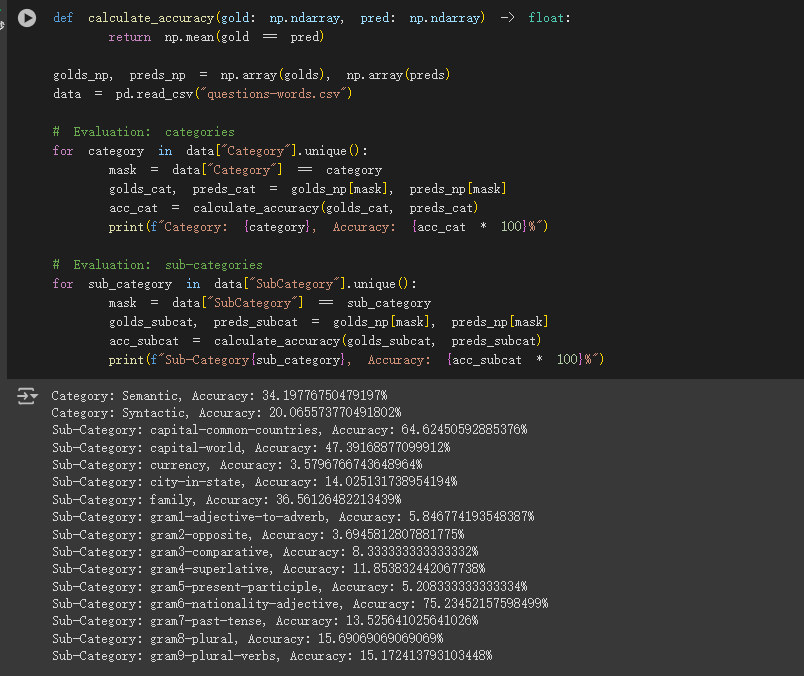
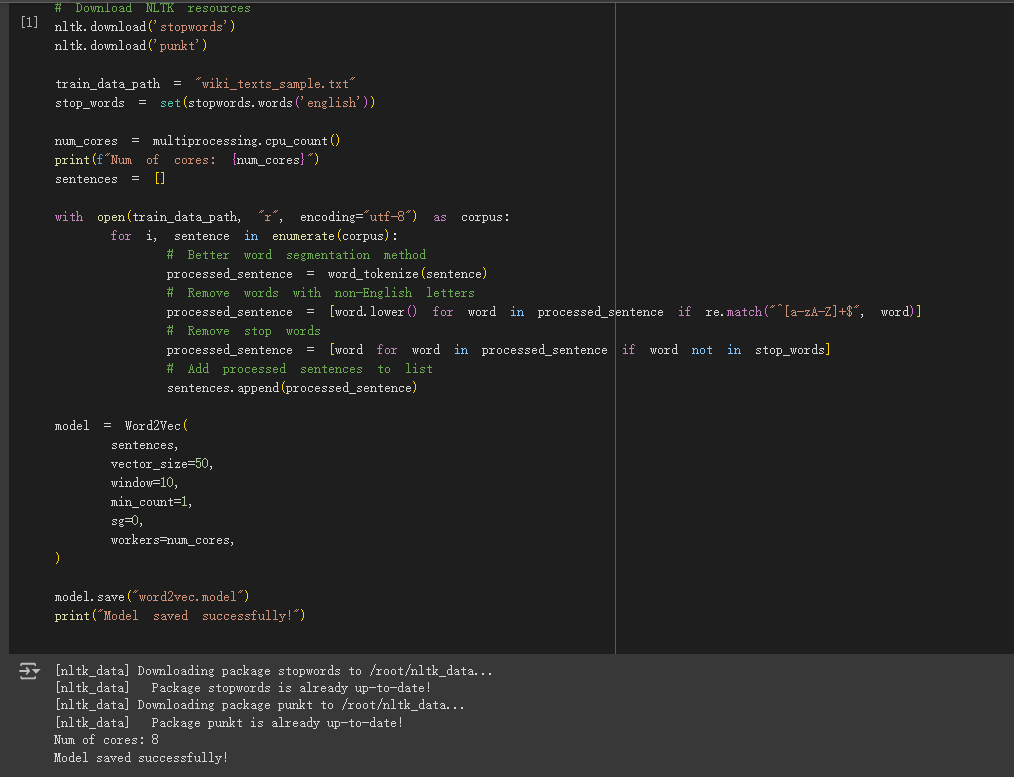
2.

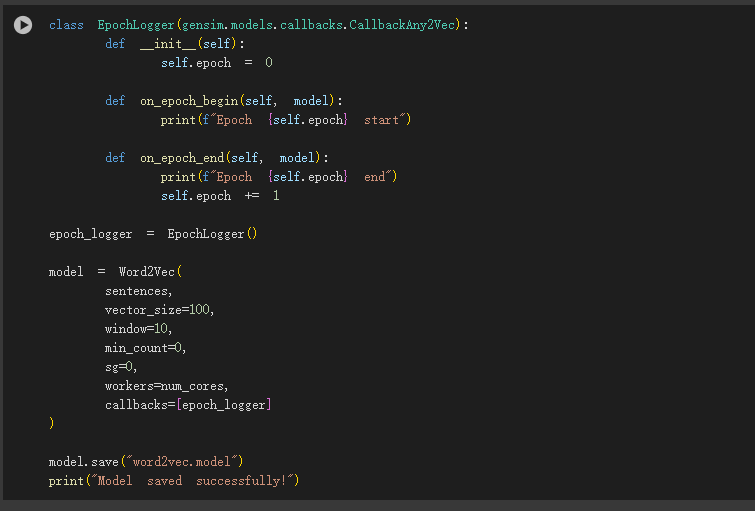


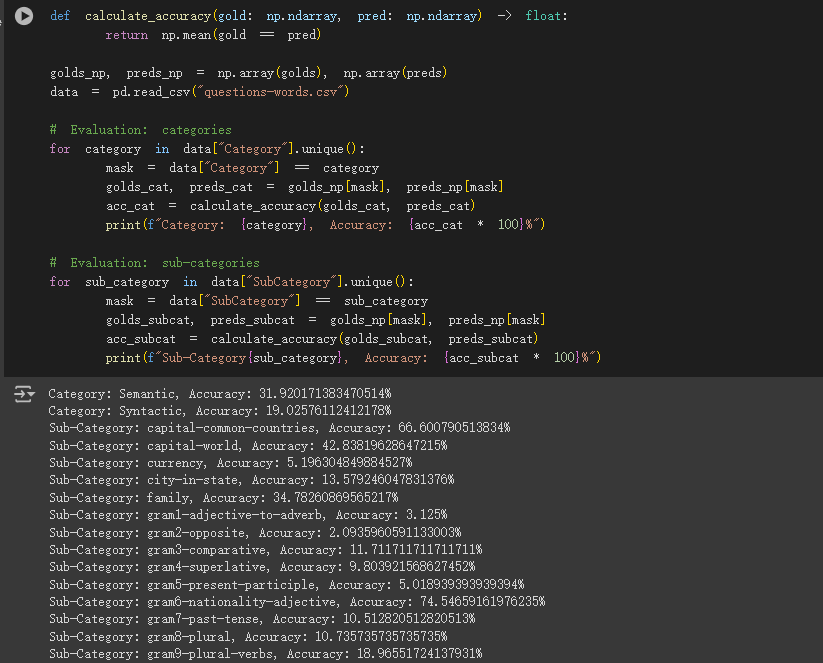
3.



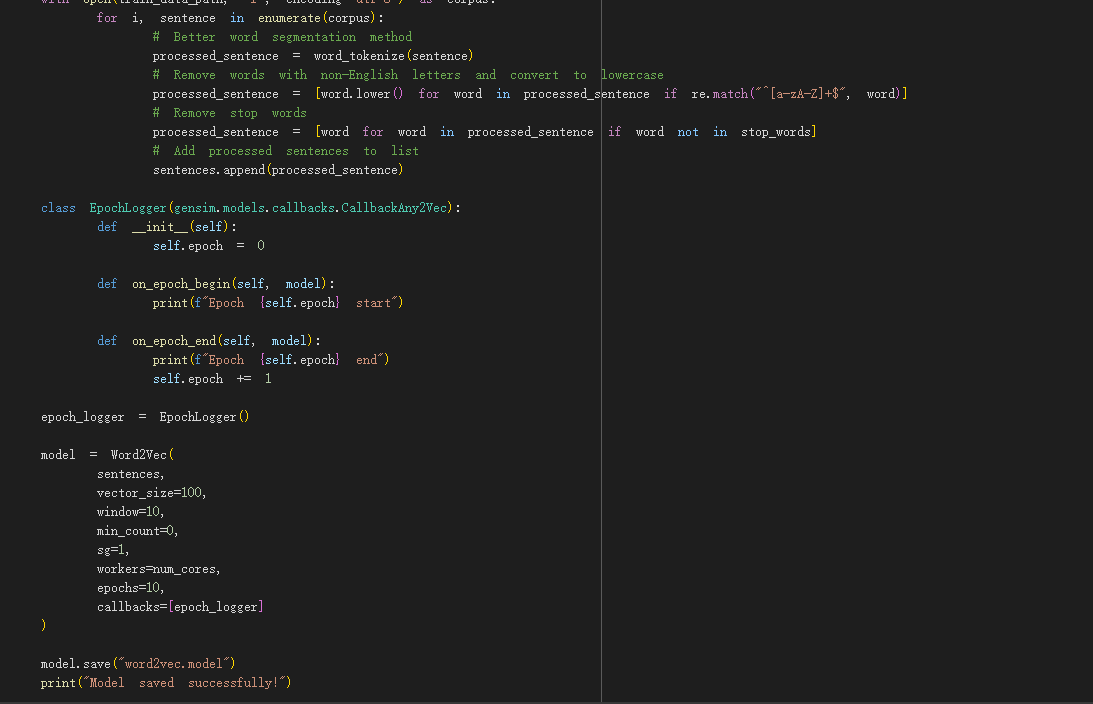
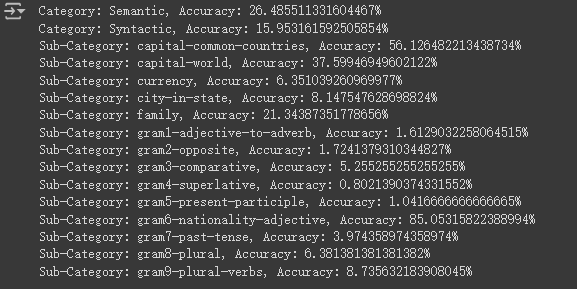
4.



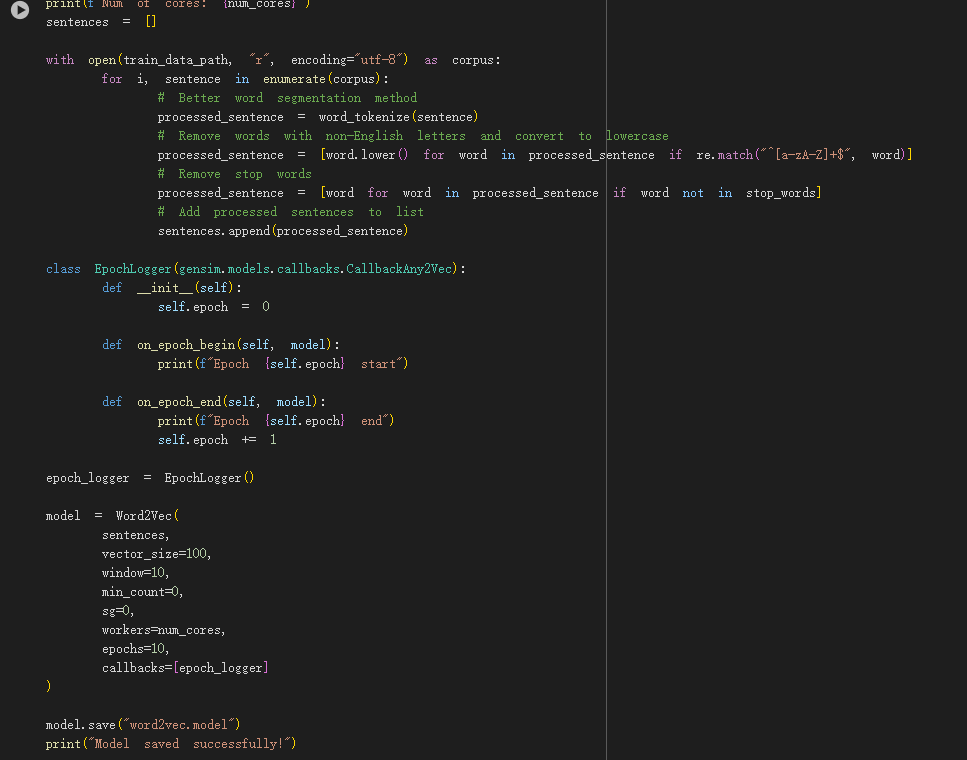
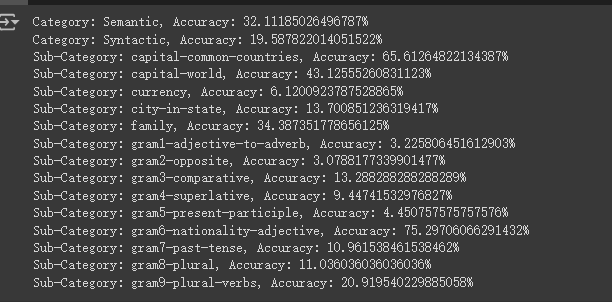
5. 



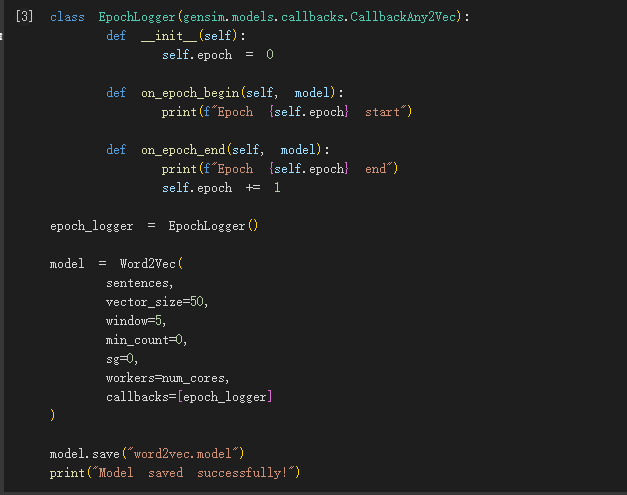
6.

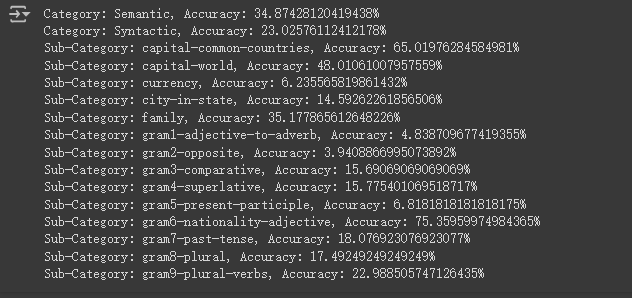
 

7.

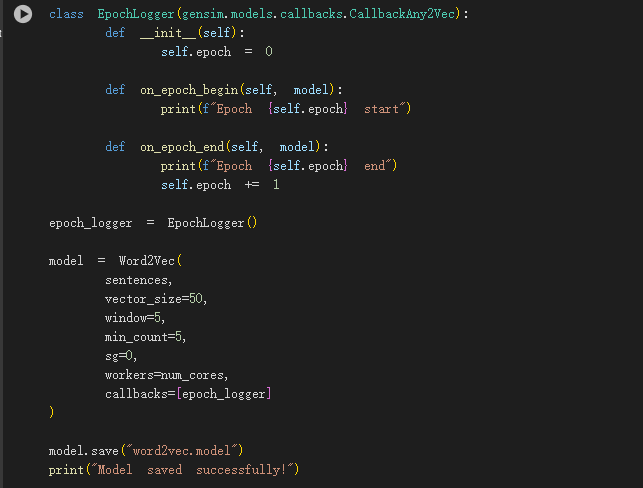
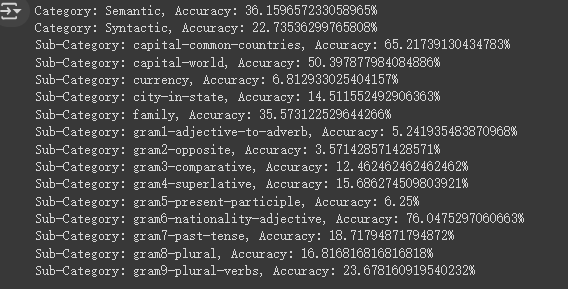
 

8.

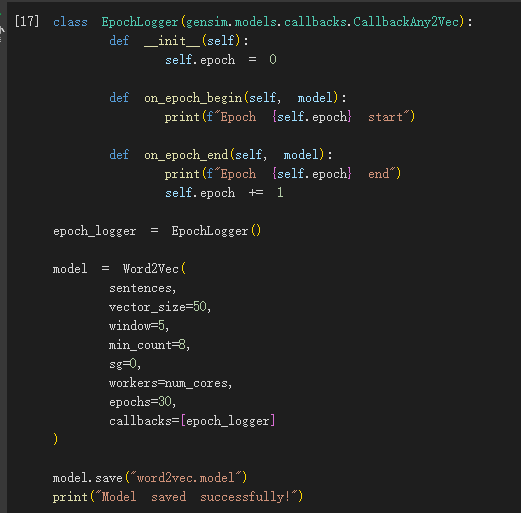


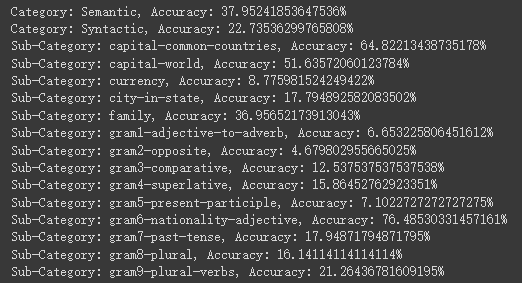


9.

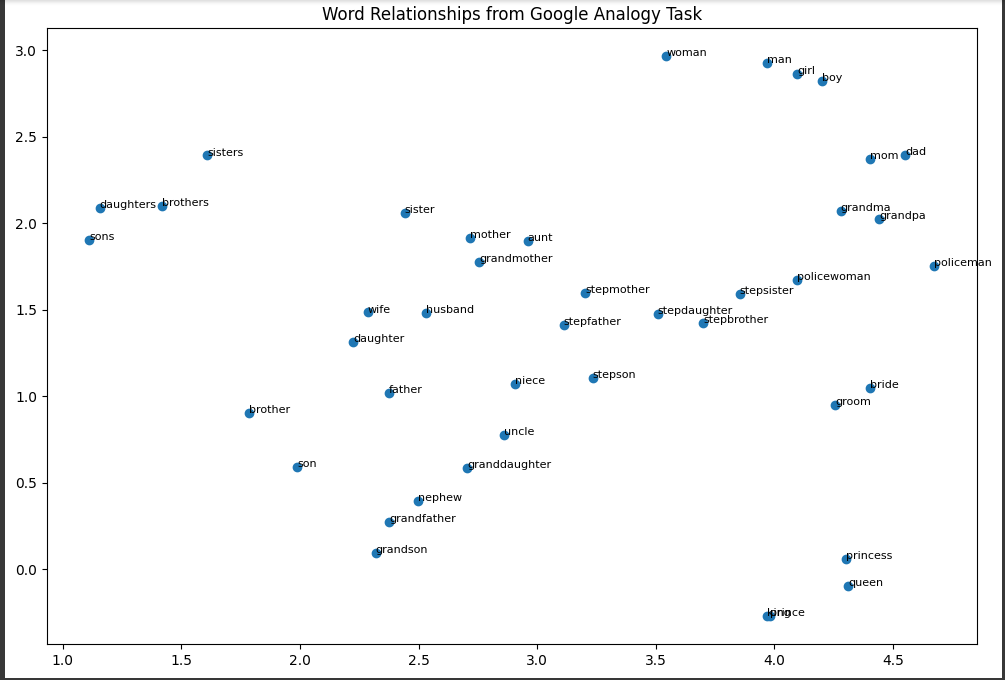
10.





It can be seen that the last parameter combination has the best effect, and its prediction accuracy is better than glove-wiki-gigaword-300

#TODO7 Plot t-SNE to see word relationships in the sub-category of `family`



II.Analysis:

● Which embedding model do you use? What are the pre-processing steps? What are the hyperparameter settings?

I use the Word2Vec model, and the data pre-processing steps include removing non-English words and stop words, and tokenizing the sentences. The reason for not doing lemmatization is that if all words are changed back to their original form, it will greatly affect the syntatic word analogy.

I have tried many settings of hyperparameters:

vector size: When vector size is set to 300, it performs poorly. It is different from what the teacher taught in class. However, I think this result may be due to insufficient training volume. Although 1 million articles seems enough, for thousands of Single words still need to use a larger amount of data to achieve better performance in multi-dimensional vectors. So I tried other smaller numbers like 50 or 100 and obviously got better performance.

window size: The window size will consider the context. If it is set too large, there will be too many reference words and sentences, making the results inaccurate. If it is too small, the context will be unclear. Settings 5-10 will have the best effect.

min\_count: Reducing low-frequency words helps speed up training and filters out noisy words. I have set it to 3 or 5 during implementation, and the result is not much different.

sg (Skip-gram / CBOW): Skip-gram performs better on smaller data sets and is suitable for capturing the semantic relationships of rare words; CBOW is suitable for large data sets and is faster to train. The effect of the CBOW is usually slightly better for big data.

workers: Number of parallel threads used. I use the number of maximum cores.

● What is the performance for different categories or sub-categories?

Overall, semantic analogies perform better than grammatical analogies, but this depends on the model and hyperparameter settings used. For example, if all adverbs and adjectives are extracted for training, it may perform well in the grammatical analogy task of converting adjectives and adverbs. In addition, small window sizes seem to perform better for grammatical analogies.

Among all semantic analogies, it can be seen that the capital-common-countries question performs best. It is conceivable that the reason is that these words often appear in the text, making the semantic positioning more precise, while currency performs the worst because of the frequency of the words.

The grammatical analogy generally performs poorly, probably because Word2Vec is a distributed representation model based on context prediction, and it does not explicitly handle the morphological changes of words. For example, "run" and "running" are independent words in the model, rather than being related by rules. If there is no significant difference in the contextual distribution of present participles and base verbs in the training data, the model may not be able to capture their morphological relationships. Only the nationality-adjective question is better. I think it captures the lexical relationship between a country and its people, rather than the grammatical relationship between nouns and adjectives.

● What do you believe is the primary factor causing the accuracy differences for your approach?

There are many factors that will greatly affect the model accuracy, including vector size, training corpus volume and data pre-processing.

The impact of vector size has been mentioned above. The main reason is that the amount of training this time is small, and using an excessively large vector size makes it difficult to express the relationship between words.

The other part is the amount of training, which actually has the most direct impact on the results. The increase in training amount has the most direct improvement in the quality of word processing tasks.

Data pre-processing has a similar concept. For example, the deletion of stop words can filter out noise and impurities, which has good performance for overall training.

● What’s your discovery from your t-SNE visualization plots?

It can be seen from the t-sne chart that the model has grasped the relationship between family-related words. For example, mom and dad cluster together, while bride and groom form another cluster. And the direction of formation of men and women is roughly the same. For example, the direction of king and queen is the same as the direction of prince and princess.

Although the prediction accuracy in the family category is not high, which may be due to too much impurity in the text data, it can be clearly seen from t-sne that the model has learned the relationship between words.

● What’s the difference in word representations if you increase the amount of training data?

With more data, the model is exposed to a wider variety of contexts for each word, leading to more accurate and nuanced word embeddings. Words will appear in different contexts, which helps the model capture a richer semantic meaning.

Words that were previously underrepresented in smaller datasets will have more occurrences, improving their embeddings. This is particularly important for less frequent words, as their representations become more robust with more training data.

● Why word2vec has poor overall effect on syntatic word analogy?

The primary objective of Word2Vec is to capture semantic similarity between words based on their context, not syntactic relationships. It focuses on predicting words from their surrounding context, making it more suited for tasks like finding semantically similar words (e.g., "king" and "queen"), rather than learning transformations like adjective to adverb or pluralization.

And both architectures of Word2Vec—Skip-Gram and CBOW—are designed to model semantic relationships. Skip-Gram tries to predict surrounding words given a target word, and CBOW predicts a target word given its context. Neither of these architectures explicitly models syntactic rules or transformations, making them less effective for syntactic analogies.