ProQuest Sentiment Analysis Final Report

Written by: Taylor Murray
Project team: Pranay Shah, Vishnu Nair, Arun Annamalai, Eamy Mo, Rakshit Gogia, Sebastian Jin, Taylor Murray
Sponsor Partners: John Dillon and Dan Hepp
Advisor: Sugih Jamin

1. Project Description and Impact

This project was in collaboration with ProQuest, an education technology company. Their main service is in providing textual content for their users, who are mostly academic researchers. This project's purpose was to enhance their users capabilities in their new text-data-mining (TDM) environment where their users can go to run data analytics on any ProQuest content they want, and also to improve ProQuests internal use of their mass amounts of textual data.

The two focal points of this project are word embeddings and sentiment analysis. ProQuest wanted custom word embeddings made on their own corpora as well as a system for their users to be able to create their own custom word embeddings. The pre-trained word embeddings could be used on any number of internal projects to improve their recommender systems, support search features and optical character recognition, and any other natural language processing tasks that the company works on. Additionally, word embeddings can be utilized for sentiment analysis, i.e. identifying emotion in text. Thus the most crucial role of the word embeddings was to train them on internal ProQuest corpora so that they could them be used when training sentiment analysis models. The hope is that training sentiment analysis models with word embeddings that themselves were trained on the same data as the sentiment task would yield better results than using word embeddings trained on more general corpora like text from Wikipedia.

In addition to sets of custom word embeddings and a word embeddings generation system, we were also tasked with creating these custom corpora sentiment analysis models along with walk-through notebooks that would allow users to predict emotion on their own data. The sentiment analysis models besides being used in the notebooks could be used by the company to help show researchers a variety of perspectives (using sentiment as a proxy) on different topics.

2. Background

Word embeddings are essentially representations of words in a vector space and are necessary for sentiment analysis. They are created using unsupervised machine learning, so though we don't know exactly what properties are being identified about the words we do know that they encode a mix of semantic and syntactic properties. They can be used in a variety of different contexts however for our purposes, we utilized them solely for sentiment analysis.

Sentiment analysis is the process of computationally identifying sentiments expressed in a document - it is powerful in the study of affective states. To obtain sentiment analysis models, i.e. functions that provide a predicted emotion when given text, we use supervised machine learning. Supervised machine learning means we have to provide sentences with emotion labels so that the model can start to identify what words correspond to each emotion.

We were given three datasets by ProQuest to utilize: LION (Literature Online) poetry, Book Blurbs, and NYTimes Articles 1960s-2018. LION poetry is a set of 389,000 poems, Book Blurbs is a set of more than 10 million excerpts from various books and NYTimes articles is a set of more than 6 million New York Times articles that includes the following categories: Obituary, Op-ed, Feature, Article, General Information, News, Front page/cover story, Correspondence, Letter to the Editor, Review, Commentary, and Editorial. All of these sets were used to make word embeddings, and subsets of the NYT articles and LION poems were truthed or labeled by our team with emotions so that we could use them for sentiment analysis tasks as well.

When doing truthing we randomly chose the documents and labeled each sentence with one of nine emotions that best described the emotion of the writer. We used Ekman's 6 emotions [1]: *anger, disgust, fear, sadness, happiness,* and *surprise* and we added on 3 extra emotions: *love, neutral,* and *other*. We added *love* because we were expecting it to be a dominant emotion in our poems dataset and we believed it was distinct enough from the other emotions. We also added *neutral* in the case that the author wasn't expressing any emotion, which we also expected to see for a lot of the New York Times data, given that their goal is to objectively report the facts. And finally we added *other* in the case that a sentence did not exhibit any of the other emotions.

3. Sentiment Datasets

This section gives an overview of our 4 datasets we use for sentiment analysis tasks: NYT, LION poems, SemEval, and Fairytales, as well as our dataset for valence analysis: Stanford Treebank.

.

3.1 NYT and LION poems Datasets

These datasets are the two datasets our team created ourselves by each member truthing a random subset of the LION poems and NYT data given to us by ProQuest. We truthed 1,132 sentences for the LION poems dataset and 1,852 sentences for the NYT dataset. Both have 9 possible emotion labels: *anger, fear, disgust, happiness, sadness, surprise, love, neutral,* and *other*. They are composed mainly of neutrals. Below you can see both datasets' emotion distribution:

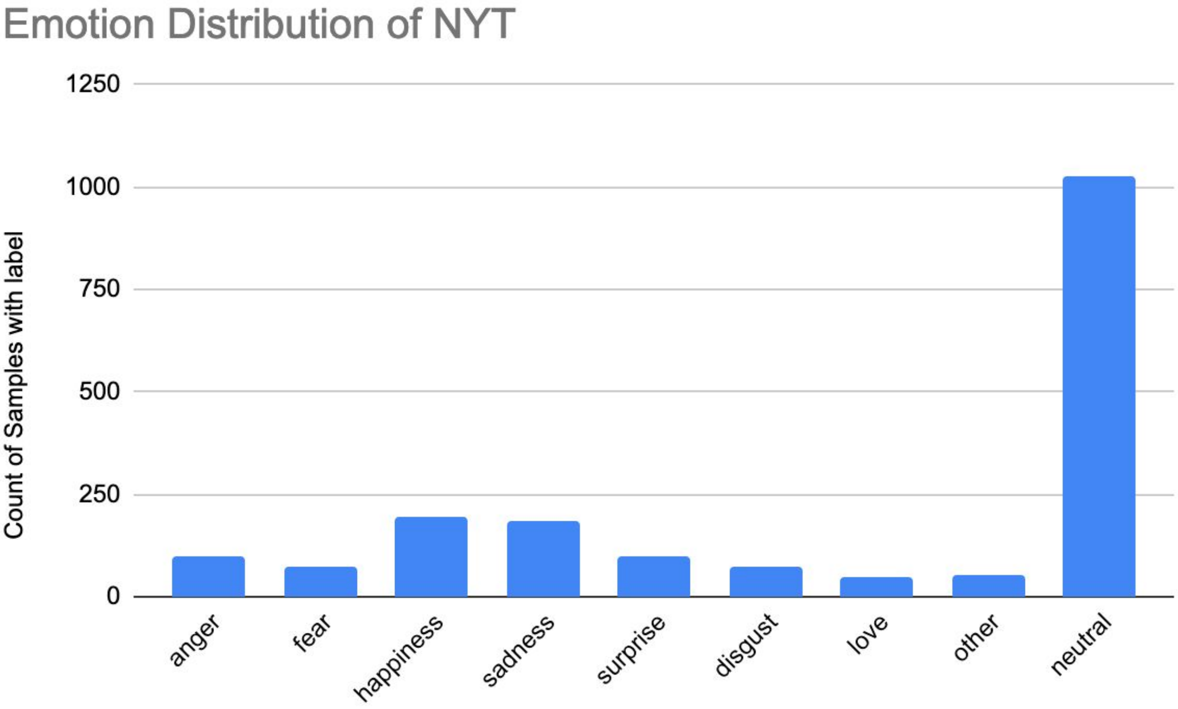

Figure 3.1 - Distribution of each emotion label in the New York Times truthed dataset

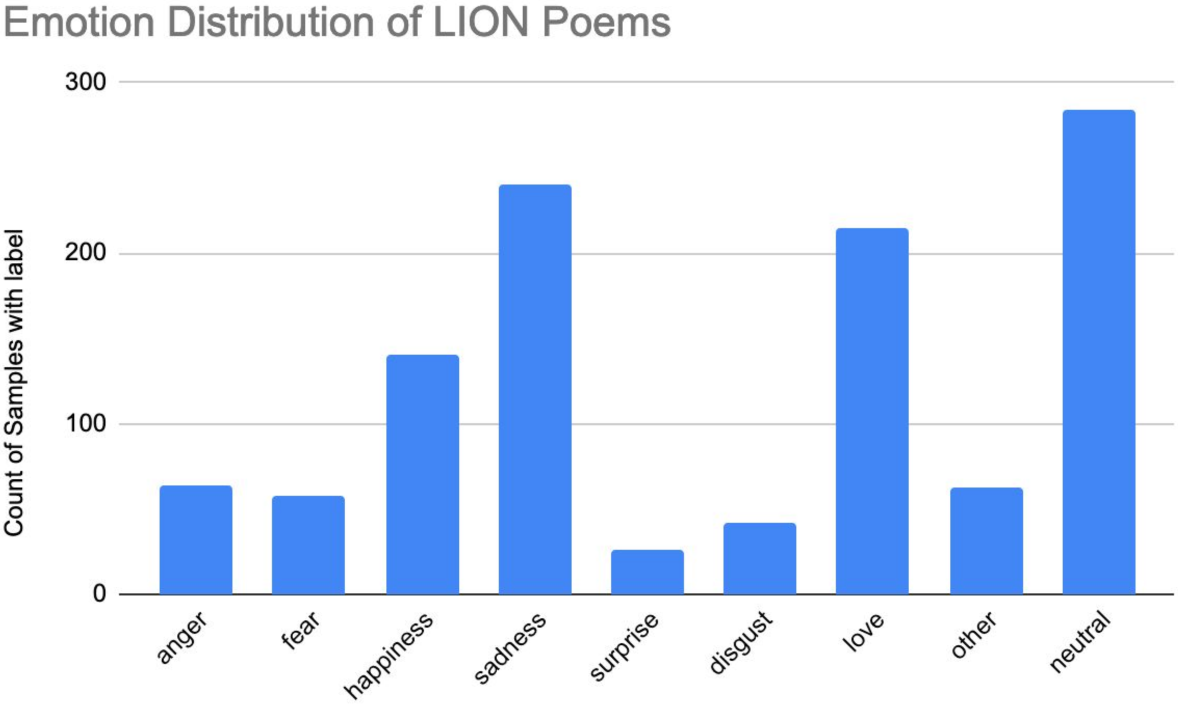

Figure 3.2 - Distribution of each emotion label in the LION poems truthed dataset

3.2 SemEval-2007 Dataset

The SemEval-2007 dataset was created by Carlo Strapparava and Rada Mihalcea and it is composed of 1250 news headlines [2]. For each headline there are 6 reported scores for each emotion, indicating how much human evaluators felt that that emotion was present in the headline on a scale of 0 (no presence of that emotion) to 100 (strong presence of that emotion). To make the labels discrete for each sentence we chose the label with the highest score as the label for that sentence. While technically there are two sets, a train (1000 sentences) and test set (250 sentences), we decided to combine them into one set for our evaluation. The SemEval dataset has only 6 emotion labels (corresponding to Ekman's 6 emotions [1]): *sadness, surprise, joy/happiness, anger, fear,* and *disgust*, making it our only dataset without neutrals. Below you can see the emotion distribution for SemEval:

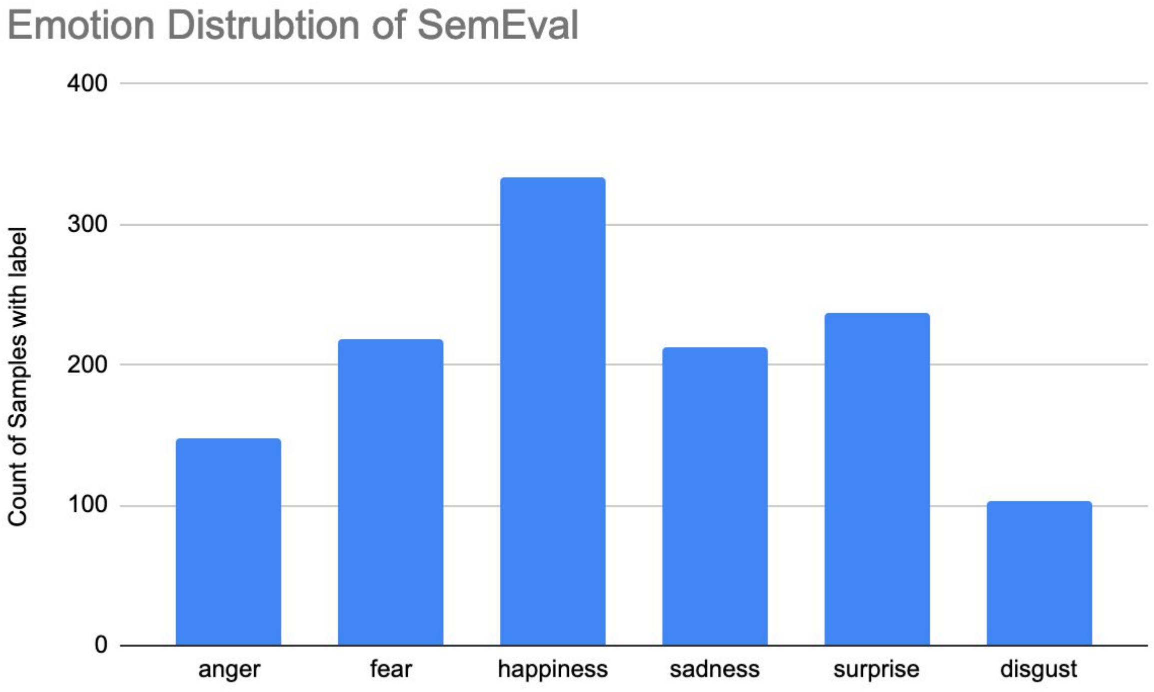

Figure 3.3 - Distribution of each emotion label in the SemEval dataset

3.3 FairyTales Dataset

The fairytale dataset is a dataset created by Cecilia Ovesdotter Alm and is composed of 15,302 sentences from fairytales by authors such as Beatrix Potter, H.C. Andersen and Grimms [3]. There is only one label per sentence given by a human annotator. The dataset has seven possible emotion labels: *neutral, angry, fearful, happy, disgusted, surprised* (technically it contains surprise positive and surprise negative but we combined them into one surprised), and *sad*. Below you can see the emotion distribution for FairyTales:

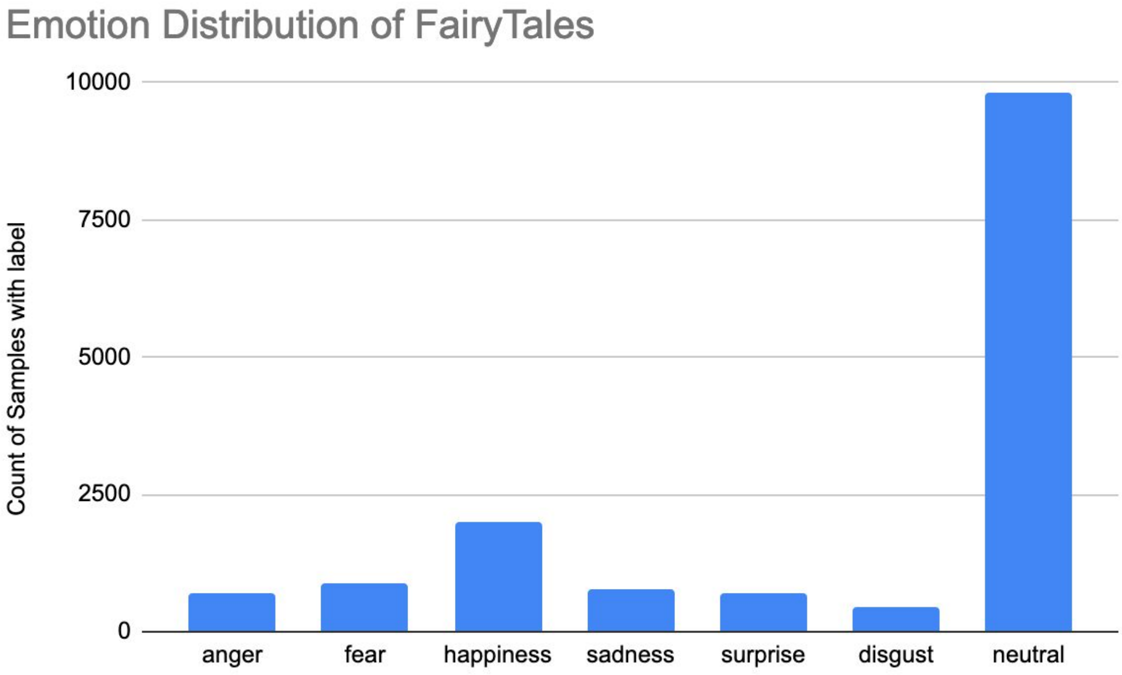

Figure 3.4 - Distribution of each emotion label in the FairyTales dataset

3.4 Stanford Treebank

Our last dataset is our only valence set which measures how positive or negative a text is and was created by a team at Stanford including Richard Socher, Alex Perelygin, Jean Y. Wu, Jason Chuang,Christopher D. Manning, Andrew Y. Ng and Christopher Potts [4]. It consists of 10,754 sentences extracted from movie reviews. We use the fine-grained version where each sentence is given one of 5 valence labels: *very positive, positive, negative, very negative,* and *neutral.* Below you can see the emotion distribution for Stanford Treebank:

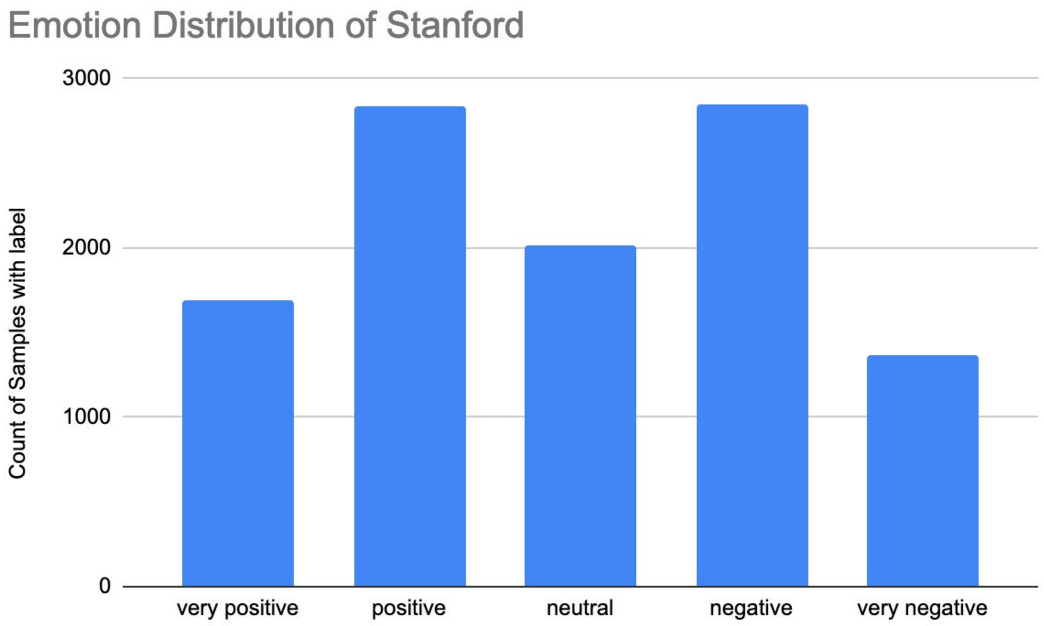

Figure 3.5 - Distribution of each emotion label in the Stanford Treebank dataset

4. Problem

Over the course of this project we were looking to identify the best sentiment analysis model for classifying affective state overall, valence state overall, and specific emotions in the hopes of proving that our sentiment analysis models trained with in-domain word embeddings (meaning word embeddings trained on the same set that the sentiment task is on) would be able to outperform models with general word embeddings.

5. Methodologies

This section goes into how we evaluated our various word embeddings and then it transitions into talking about the various sentiment analysis models used and the methodology for testing the models on each sentiment dataset.

5.1 Word Embeddings Evaluation Methodology

First we identified which word embeddings were the most promising from each dataset, i.e. the best word embeddings trained on the New York Times data, the best word embeddings trained on LION poems, and the best word embeddings trained on book blurbs. The word embeddings varied by how much data they trained on as well as whether or not they used an epoch method to train instead of one-pass.

We identified the best word embeddings by using 3 types of intrinsic evaluators: relatedness/similarity, analogies, and concept categorization. Intrinsic evaluators are intrinsic because they are not performing some downstream application, they are simply measuring the innate relationship between the vectors.

5.1.1 Intrinsic Tests

Relatedness tests compare how well word embeddings encode the relationship between pairs of words (as determined by human evaluators). The test sets consist of pairs of words with their relatedness rating. After obtaining the word embeddings for both of the words we measure the vector similarity by computing their cosine similarity. To evaluate, we simply take the spearman correlation coefficient between our cosine similarity scores and the human evaluator's scores.

Analogies tests evaluate the degree to which the word embeddings encode similar relationships between pairs. Suppose we have representations for two pairs of words (a1, b1) and (a2, b2), both having an analogous syntactic or semantic relation: a1 is to b1 what a2 is to b2. By the word analogy assumption, this analogous relation should be represented in terms of some optimal vector r: $r \approx b1 - a1 \approx b2 - a2$. The typical example used is the pair (man, king) and (woman, queen), where the optimal vector would be some common vector r = king − man = queen - woman. The analogies set contains a list of these kinds of similar related pairs of word pairs and we evaluate our performance by calculating $b1 - a1 + a2$ to see whether the closest word embedding is the embedding for b2.

Concept Categorization (also called *word clustering*) evaluates the relationships between word embeddings by clustering a set of words in various categories. For example, we might have 5 sets of categories (e.g. places, foods, professions, planets, etc), each with 10 words in them. We would then convert all the words into their word embedding form and run a k-means clustering algorithm on them. We evaluate the clustering by measuring the purity score [5].

You can find the specific tests we used for each intrinsic evaluator category in Appendix A.

5.1.2 Sentiment Analysis Models

After identifying the best performing word embeddings for each dataset, we created 3 sentiment analysis models for each -- one using LSTM, one using linear regression, and one using linear regression with synthetic minority oversampling technique done to it (also referred to as SMOTE). Because our own labelled datasets were heavily dominated by the "neutral" class, we used SMOTE to add duplicate elements of the minority classes when training to remove bias from our classifier for the majority class.

The other models we looked at were XLNet, sbert-bert, and google models. XLNet is a pretrained sentiment analysis model trained with BooksCorpus, English Wikipedia, Giga5 news articles, ClueWeb and CommonCrawl web crawls. There are 6 SBERT-BERT models:

- **Sbert-bert-base-nli-max-tokens**
- **Sbert-bert-base-nli-mean-tokens**
- **Sbert-bert-base-nli-stsb-mean-tokens**
- **Sbert-bert-base-wikipedia-sections-mean-tokens**
- **Sbert-bert-base-nli-cls-tokens**
- **Sbert-bert-large-nli-stsb-mean-tokens**

Each of these models starts by getting the word embeddings for each word in a given sentence using a BERT model and then uses Sentence-BERT, shorted as SBERT, to create a sentence embedding from those word embeddings. It has 3 techniques for creating sentence embeddings: max (where you compute a max-over-time of the output vectors), mean (where you take an average of the word embeddings), and simply using the CLS token that is attached to every sentence in BERT.

As for the different kinds of BERT configurations, we have 3 parameters for each: model, training dataset, and fine-tuning dataset. There are two models: base and large. The **base** model is a 12-layer, 768-hidden, 12-heads, 110M parameter neural network architecture, and the **large** model is a 24-layer, 1024-hidden, 16-heads, 340M parameter neural network architecture. And we have 2 different training sets: **nli** which means the model was trained on the Natural Language Inference Corpus (SNLI) and the Multi-Genre Natural Language Inference Corpus (MultiNLI), **wikipedia-sections** which means the model was trained on English Wikipedia, and **stsb** which means the model was trained on the Semantic Textual Similarity (STS) benchmark. Additionally, all of these models used a linear regression machine learning model when fine tuned again on each sentiment dataset. Finally, the Google model is a linear regression model using the default embeddings given by Gensim.

We evaluated our models on 5 datasets, our New York Times labelled data, LION poems labelled data, the FairyTales dataset, and the SemEval dataset, and we an aggregation of the four previous datasets, called Aggregate. To evaluate on each, we did an 80/20 split, meaning we trained on 80% of the data and tested on the other 20% for each dataset.

To determine the best sentiment analysis model overall, we took the average of the macro f1-scores of each of the 5 datasets (NYT, LION, FairyTales, SemEval, and Aggregate). And for finding the best valence model overall, we simply took the macro-f1 score on the Stanford Dataset. To find the best sentiment score for each emotion we took the average of the f1-scores for that emotion over all of the datasets that included that emotion.

6. Results

We will first dive into our results for which word embeddings performed the best for each of ProQuest's three corpora, and then we will move on to the results using those word embeddings in our comparisons on sentiment analysis tasks.

6.1 Word Embeddings Results and Recommendations

Each section describes the performance of our word embeddings on concept categorization, analogy, and relatedness tasks, for each dataset given to us by ProQuest. It should be noted that each of these word embeddings was created using the Gensim package.

6.1.1 Book blurbs corpora

First we compare word embeddings made on the book blurbs dataset, where we see **skip-gram_book-blurbs_3.5.bin** perform the best in relatedness and analogy tasks, and comparable in the concept categorization tasks. Because of these accomplishments, we chose **skip-gram_book-blurbs_3.5.bin** to be the word embeddings we used for our sentiment analysis model trained on the book blurbs dataset.

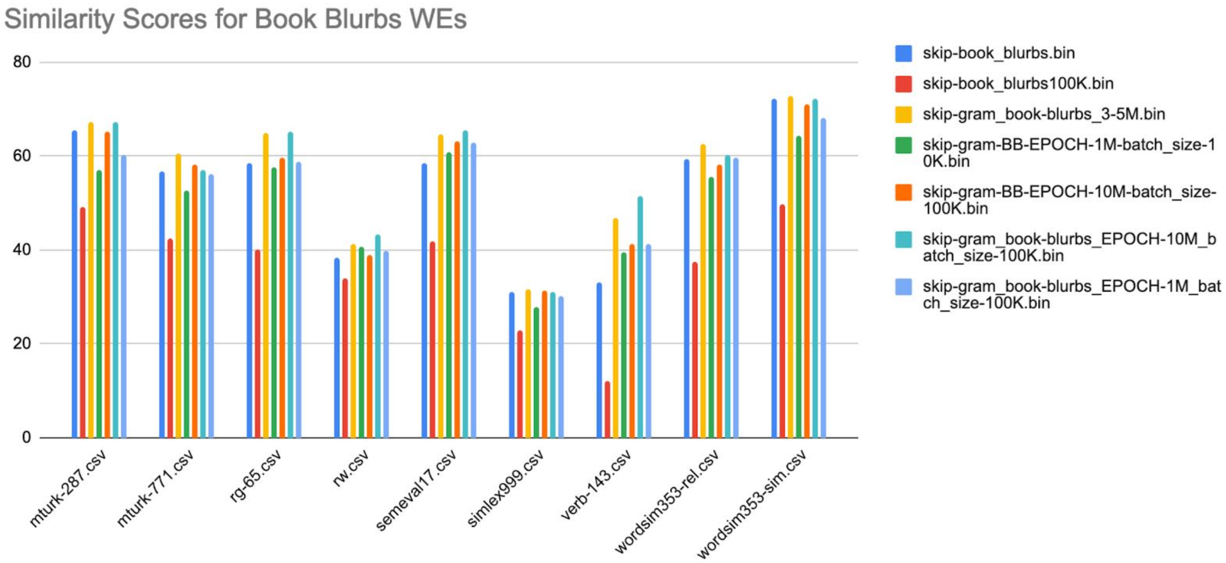

Figure 6.1 - Performance of book blurb word embedding models on similarity (relatedness) tests

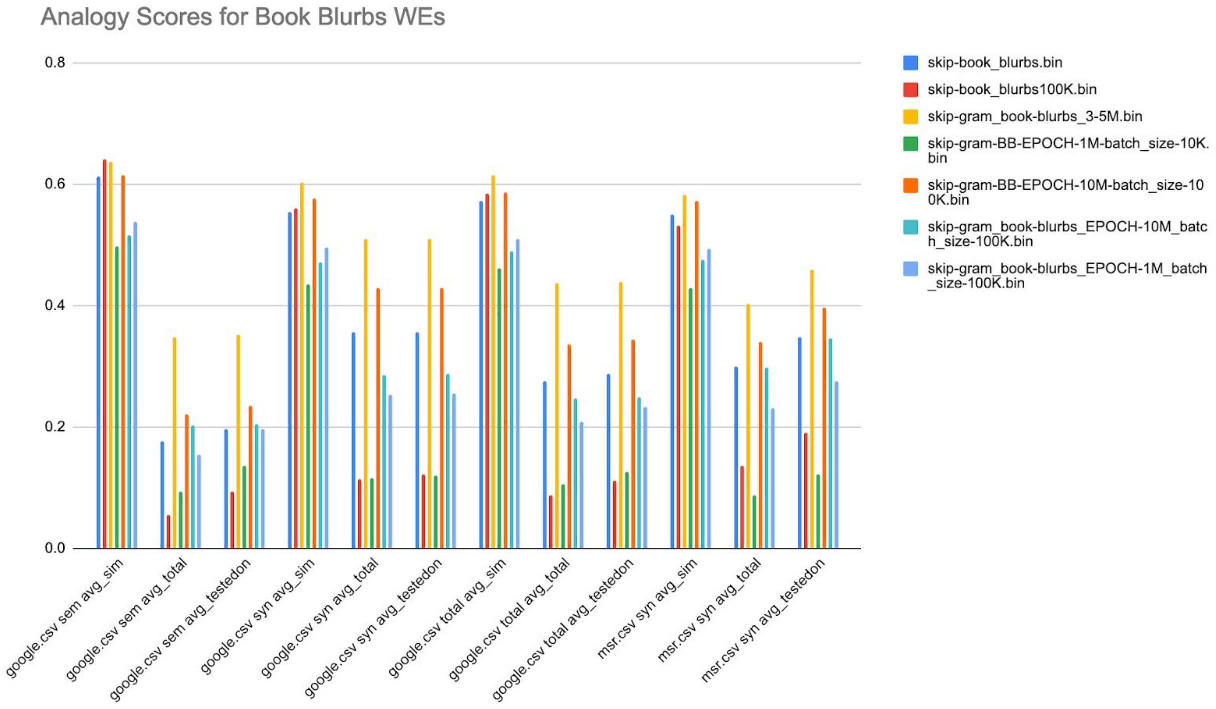

Figure 6.2 - Performance of book blurb word embedding models on analogy tests

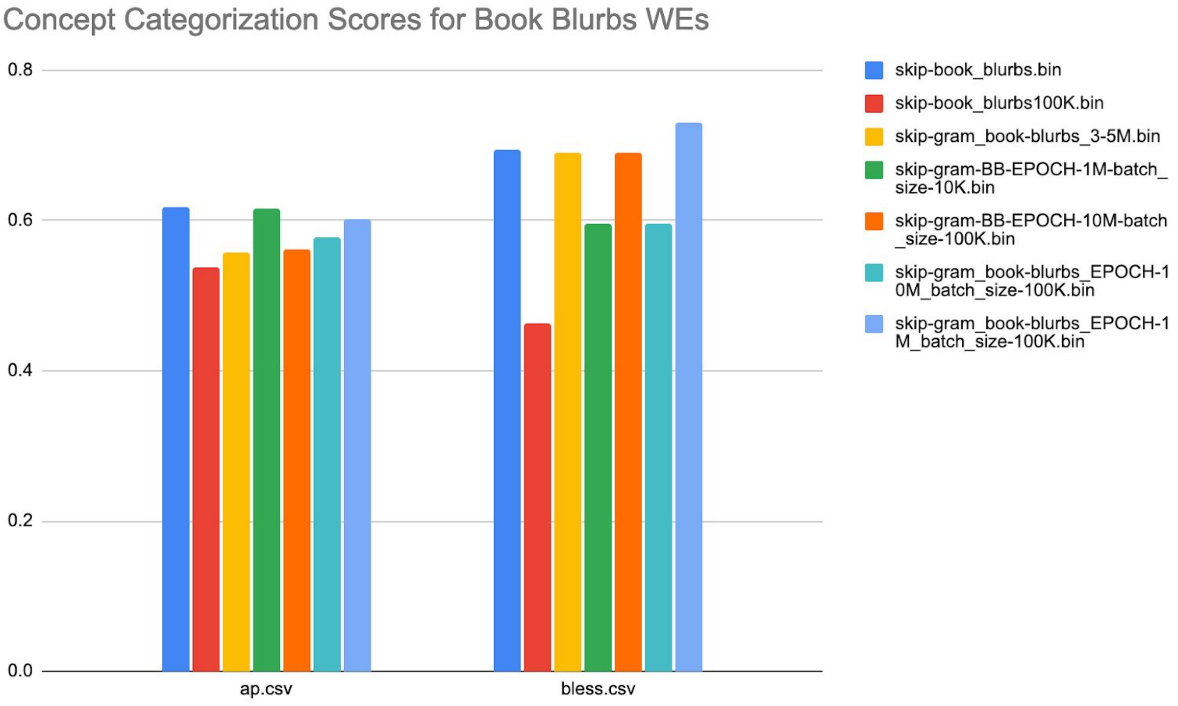

Figure 6.3 - Performance of book blurb word embedding models on concept categorization tests

6.1.2 LION poems corpora

Next we compare word embeddings made on the LION dataset, where we see **skip-gram_Lion-Poem-v2.bin** perform the best in all of the relatedness tasks and concept categorization tasks, and perform equally to **skip-gram_Lion-Poem.bin** on the analogy tasks (which is why a graph is not pictured). Because of these accomplishments, we chose **skip-gram_Lion-Poem-v2.bin** to be the word embeddings we used for our sentiment analysis model trained on the LION poem dataset.

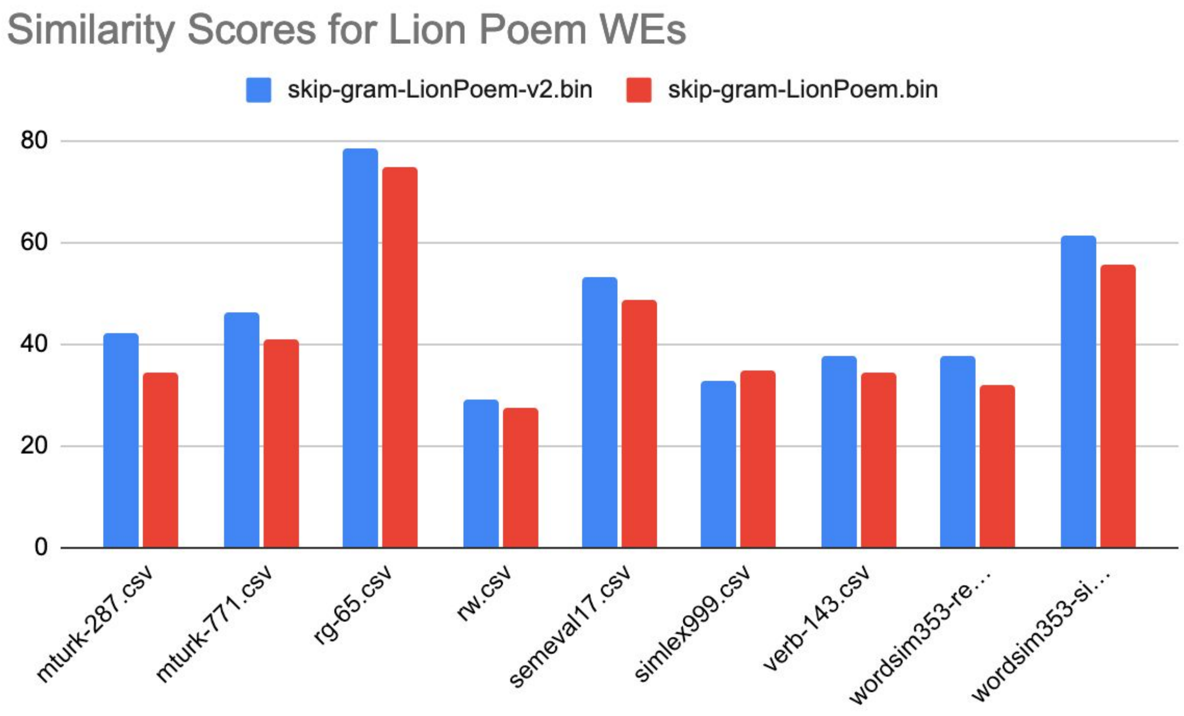

Figure 6.4 - Performance of LION poem word embedding models on similarity (relatedness) tests

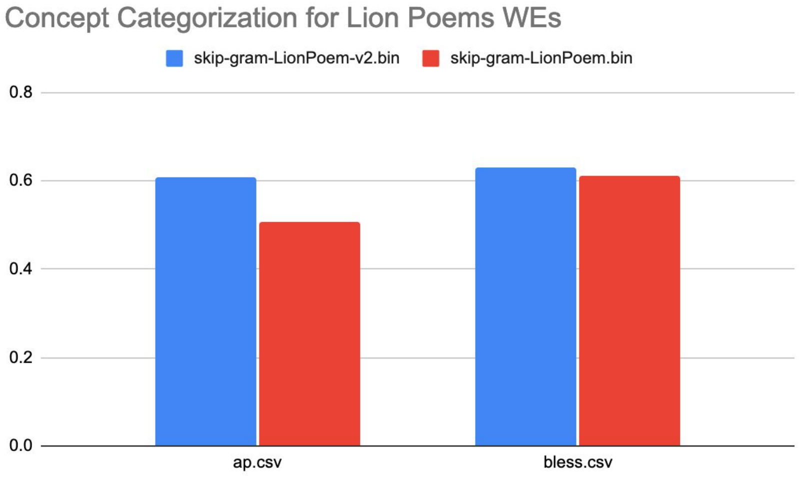

Figure 6.5 - Performance of book blurb word embedding models on concept categorization tests

6.1.3 NYT corpora

Next we compare word embeddings made on the NYT dataset. Unfortunately we were not able to test the **skip-gram_6M.bin** and **skip-gram-3M-EPOCH-documents.bin** word embeddings on the analogy tasks, however we do see **skip-gram_6M.bin** do the best overall on the relatedness tasks and comparable to the other WEs on the concept categorization task, so we have chosen it to use for our NYT word embedding representative for sentiment analysis. Another interesting result is the strong performance of the **skip-gram_2-4_256GB-mem.bin** word embeddings on the analogy tasks and comparable performance on the relatedness and concept categorization tasks. It is interesting to know by comparing **skip-gram_2-4_256GB-mem.bin vs. skip-gram_6M.bin**, there will be no improvements on its performance if adding more vocabulary to the word embedding or even get worse results.

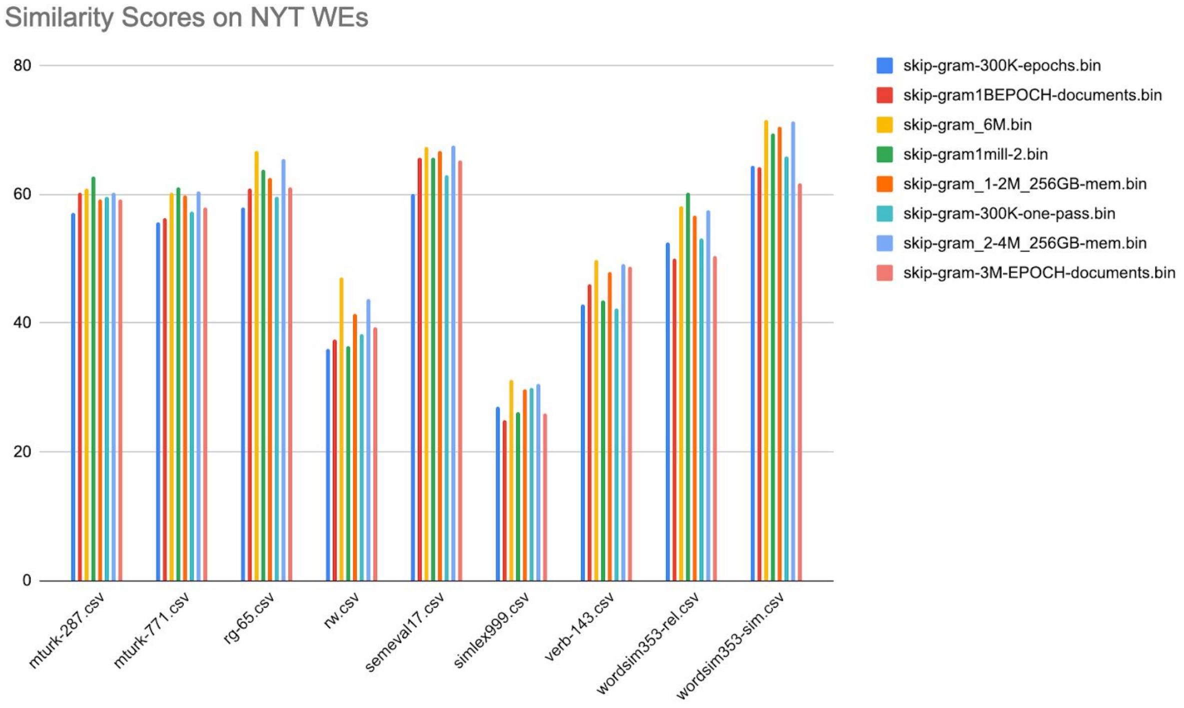

Figure 6.6 - Performance of book blurb word embedding models on similarity (relatedness) tests

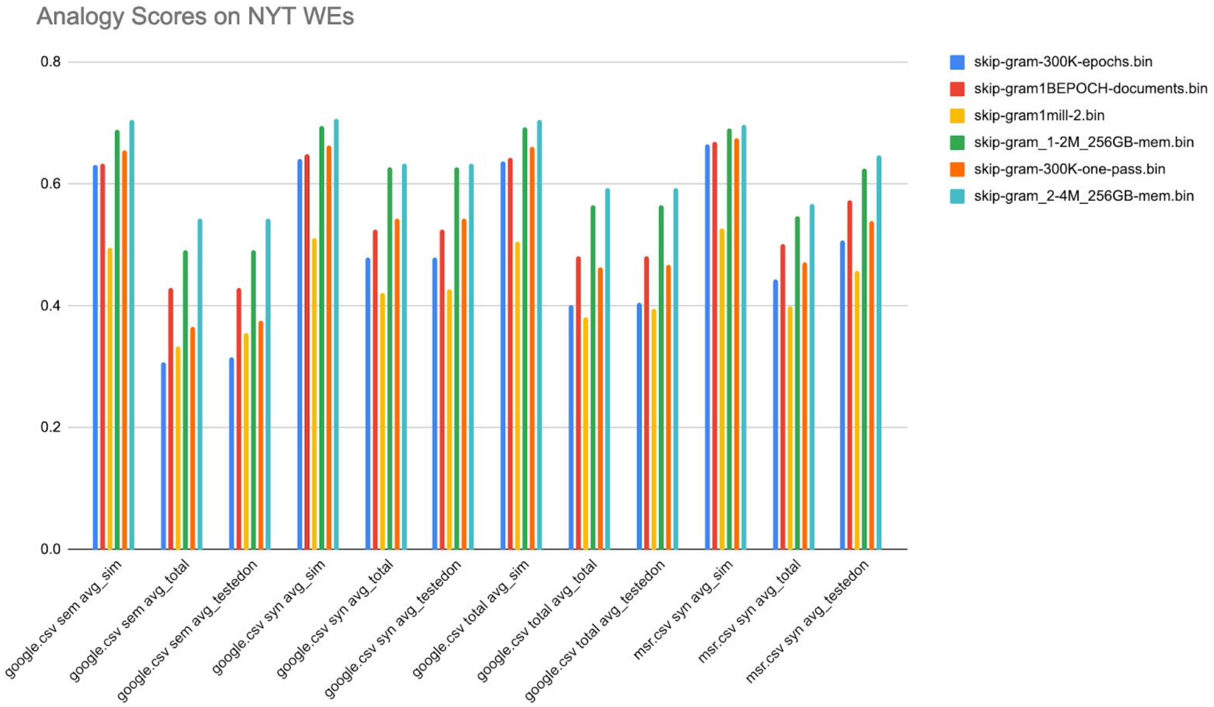

Figure 6.7 - Performance of NYT word embedding models on analogy tests

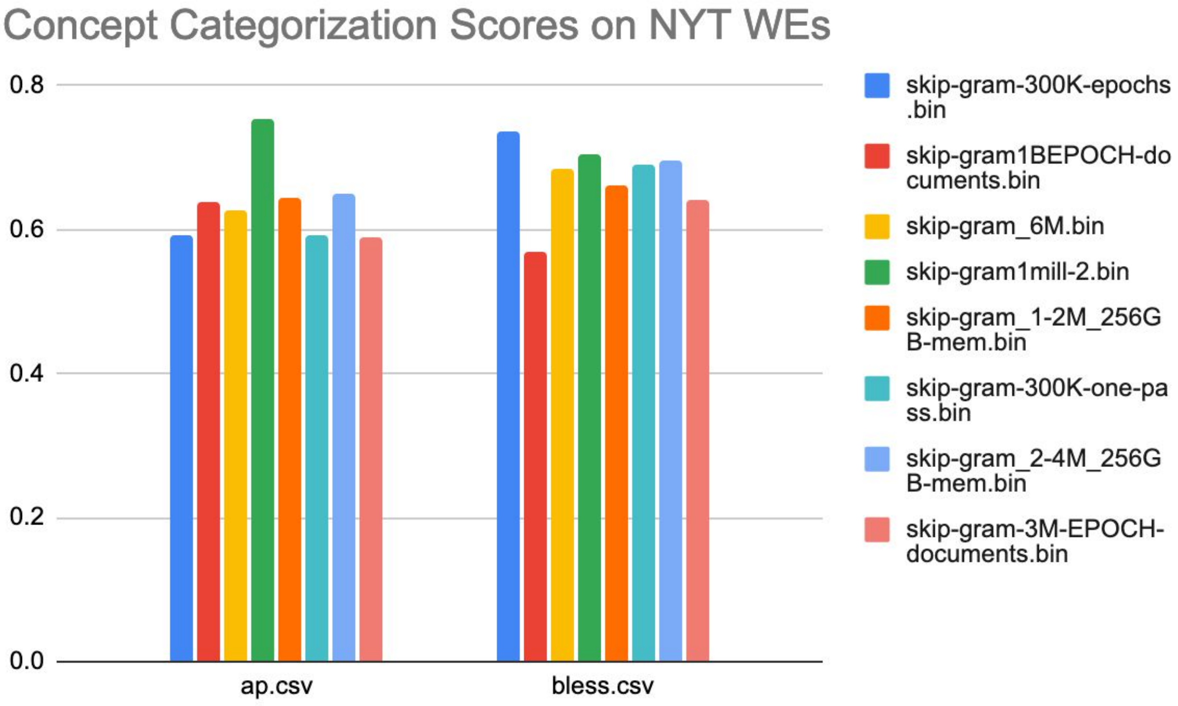

Figure 6.8 - Performance of NYT word embedding models on concept categorization tests

6.2 Sentiment Analysis Results and Recommendations

This section goes into the performance of our sentiment analysis models on overall valence performance, overall affective state performance, as well as the emotion specific performances. After giving the performance, it also gives the recommendation for which model performs the best. This model will be used in our corresponding notebook for TDM users to run on their own datasets.

6.2.1 Valence Results

Model recommendation: **XLNet** (fine-tuned on Stanford dataset)

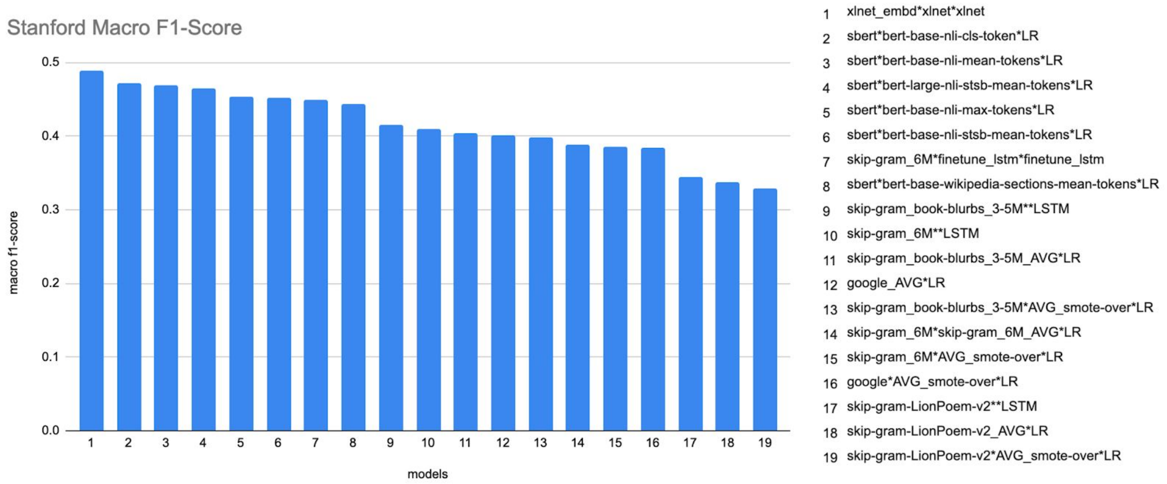

Figure 6.9: Performance of all sentiment models (Macro F1-scores) for predicting valence scores

Figure 6.9 shows our models' macro F1-scores across the Stanford valence dataset. For predicting labels across five valence score classes, XLNet (#1) outperforms all other models. The state-of-the-art (SOTA) for Stanford is a 56.2%[6] for accuracy using a Biattentive-Classification-Network (BCN) model. Our models' performance compared with the SOTA model's performance is graphed below in Figure 6.10, where you can see XLNet performs the 2nd best.

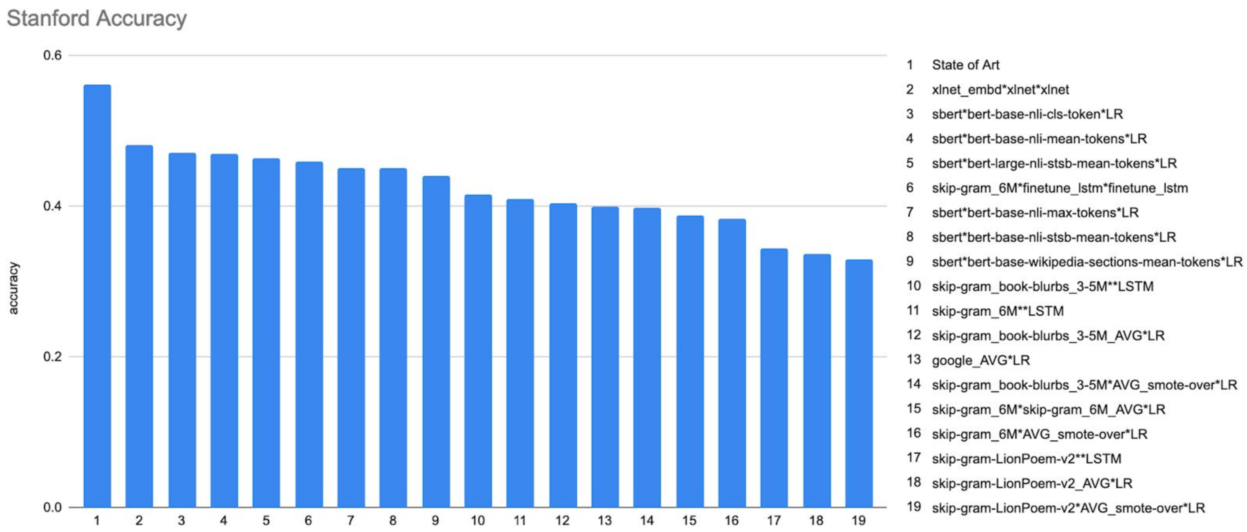

Figure 6.10: Performance of all sentiment models (F1-scores) for predicting valence scores

6.2.2 Overall Affective State Predictions

Model recommendation: **sbert-bert-base-nli-mean-tokens-LR**

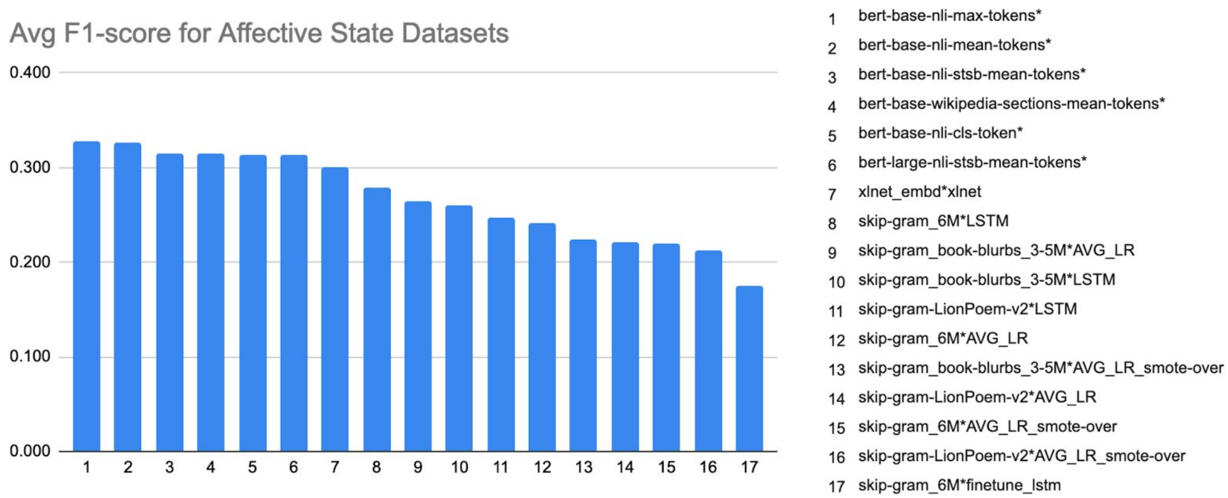

Figure 6.11: Performance of all sentiment models (F1-scores) for predicting overall affective states

For predicting labels across five valence score classes, SBERT outperforms all other models. Although the SBERT max tokens model (#1) has a higher overall performance than the SBERT mean tokens model (#2), we recommend the latter. We make this recommendation because the mean tokens model performs better at predicting more of the less common emotions like "disgust", "fear", "surprise", and "other." Figure 6.12 shows the higher performance of the mean tokens model for one of these labels. Moreover, the difference between the performance of both models overall is negligible. This allows for a more well-rounded model overall.

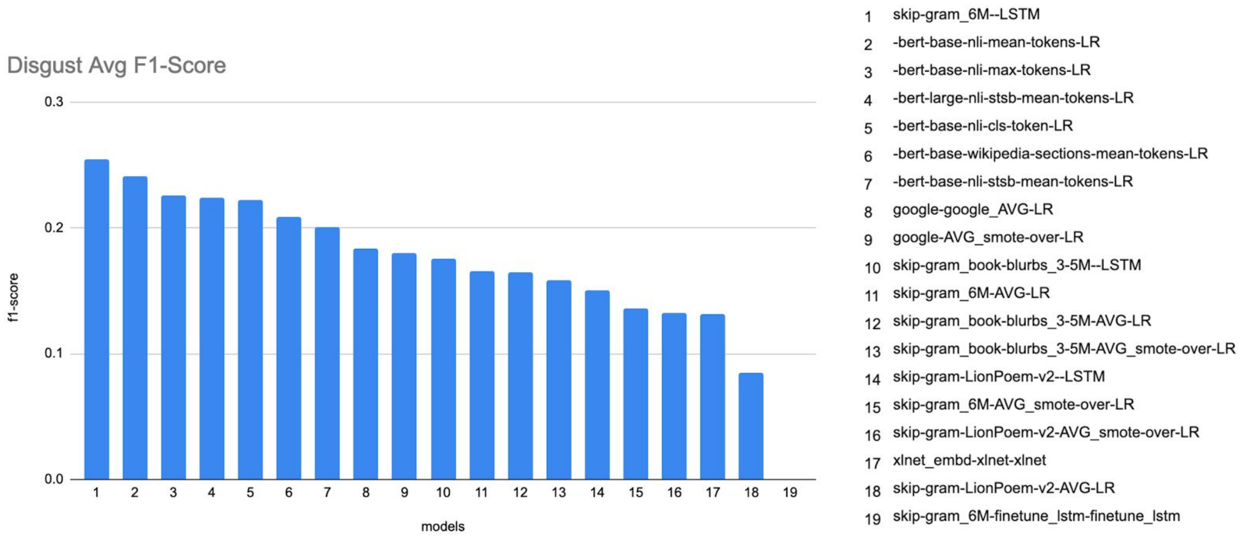

Figure 6.12: The mean tokens model outperforms the max tokens model while predicting disgust, one of the less common emotions in our dataset

6.2.3 Emotion Specific Predictions

Happiness

Model recommendation: **sbert-bert-large-nli-stsb-mean-tokens-LR**

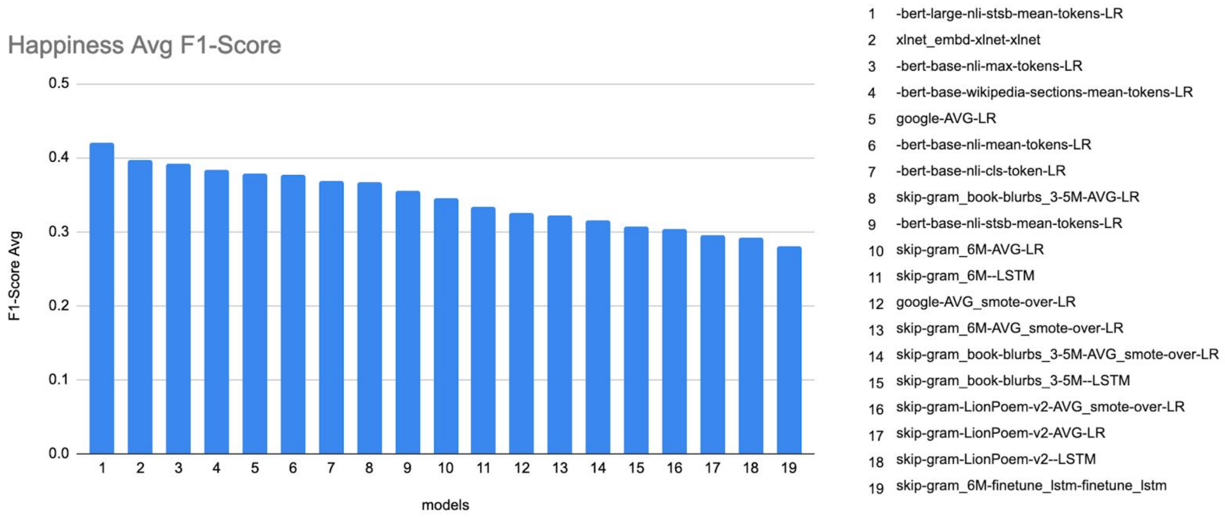

Figure 6.13: Performance of all sentiment models (F1-scores) for predicting happiness

The figure above displays the averaged F1-scores across all our models for the emotion label, happiness. On average, -bert-large-nli-stsb-mean-tokens-LR outperforms all other models for this emotion label. For happiness, the state-of-the-art F1-score across SemEval is 0.710[7]. Currently, our recommended model achieves an F1-score of .496 for happiness on the SemEval dataset.

Sadness

Model recommendation: **XLNet**

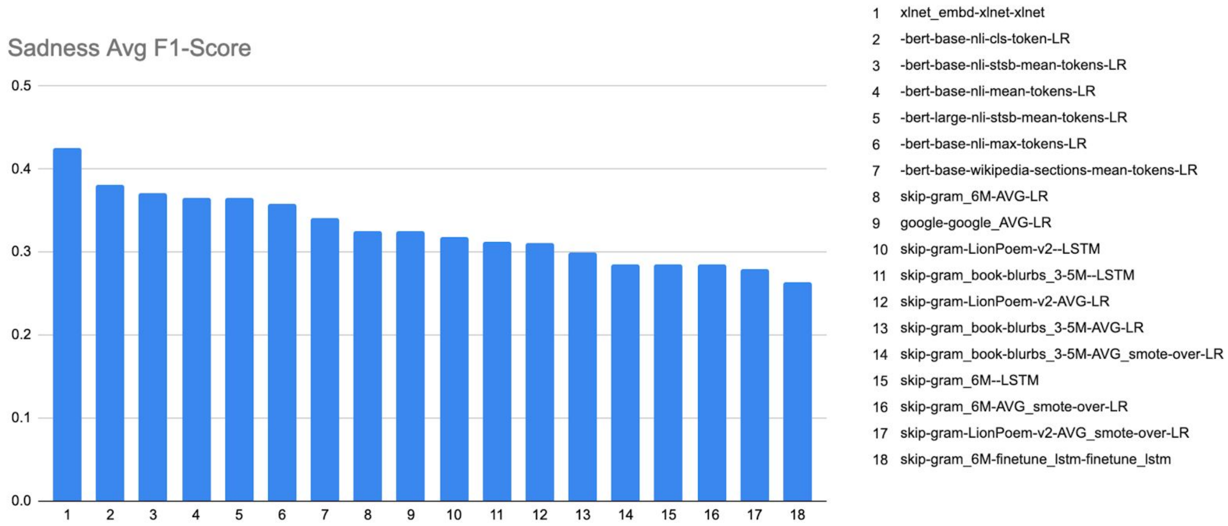

Figure 6.14: Performance of all sentiment models (F1-scores) for predicting sadness

The figure above displays the averaged F1-scores across all our models for the emotion label, sadness. On average, the average F1-score of all the trained XLNet models outperforms all other models for this emotion label. For sadness, the state-of-the-art F1-score across SemEval is 0.475[7]. Currently, our XLNet model performs slightly below that with an F1-score of .438.

Anger

Model recommendation: **sbert-bert-base-nli-max-tokens-LR**

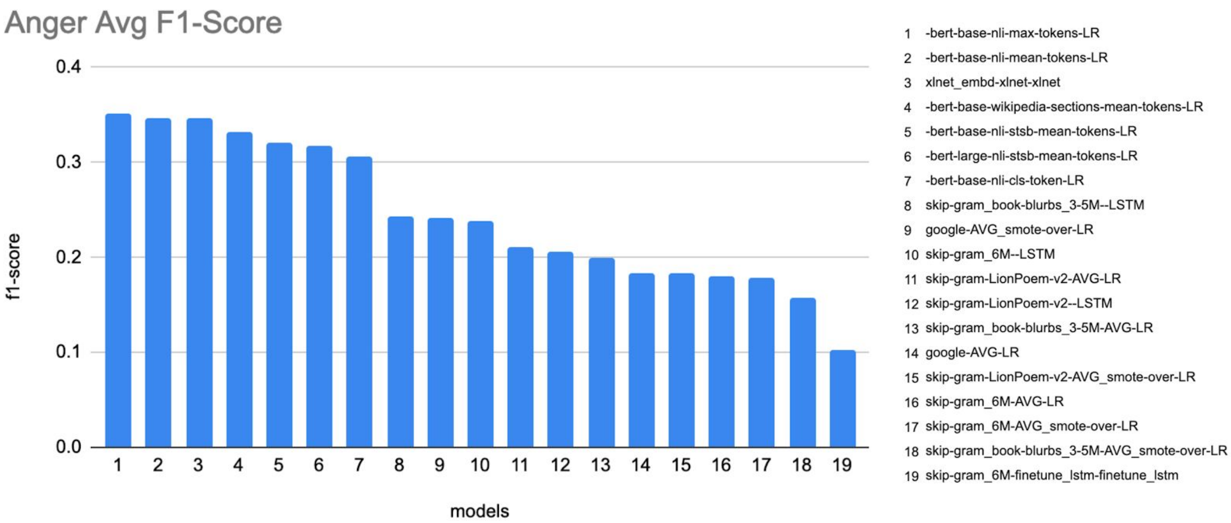

Figure 6.15: Performance of all sentiment models (F1-scores) for predicting anger

The figure above displays the averaged F1-scores across all our models for the emotion label, anger. On average, the sbert-bert-base-nli-max-tokens-LR model outperforms all other models for this emotion label. For anger, the state-of-the-art F1-score on SemEval is 0.278[7] whereas our recommended model has an F1-score of .333.

7. Discussion

One of the key questions for this project was identifying whether we could outperform external models which used general word embeddings by using word embeddings trained on the same corpora that we would then be testing on (such as NYT and LION Poems). Below you can find the macro-f1 scores for each dataset where the blue bar represents our highest macro-f1 score out of our sentiment models that used word embeddings trained on NYT, book blurbs, or LION poems, and the red bar marks the highest macro-f1 score out of the external models we used which used general word embeddings.

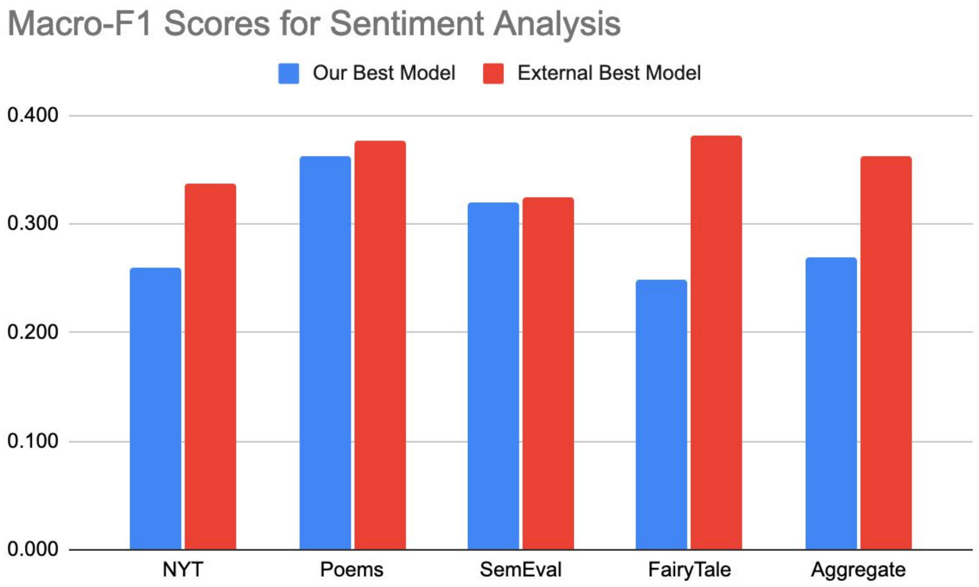

Figure 7.1: Macro-F1 scores for each dataset comparing the best external model to our best custom model

Unfortunately, it seems that in every dataset even for NYT and LION poems we see that our custom models are unable to outperform the external models. Proving at this point that just because you are using word embeddings from the same corpora to train your sentiment model, you will not necessarily have a better performance.

8. Further Exploration

One topic for further research might be into looking at sentiment as a vector, so then you could use a different kind of analysis on which sentiment analysis models performed the best. For example, if a sentence is labelled with “sadness” but we predict “anger” there should be some way to acknowledge that our model noticed a negative sentiment overall and should be rewarded for not predicting the opposite of sadness, i.e. happiness.

Additionally, it may also be interesting to see how sentiment analysis models trained with the word embeddings we did not use perform. While intrinsic evaluators like concept categorization, analogies, and relatedness give you a rough idea of the quality of your word embeddings, they don’t always correlate directly with sentiment analysis performance.

References

[1] Ekman, Paul. “An Argument for Basic Emotions.” Cognition and Emotion, vol. 6, no. 3–4, May 1992, pp. 169–200. DOI.org (Crossref), doi:10.1080/02699939208411068

[2] SemEval-2007 Task 14: Affective Text Carlo Strapparava author Rada Mihalcea author 2007-jun text Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007) Association for Computational Linguistics Prague, Czech Republic conference publication strapparava-mihalcea-2007-semeval https://www.aclweb.org/anthology/S07-1013 2007-jun 70 74

[3] C.O. Alm. 2008. Affect in Text and Speech. Lrc.cornell.edu.

[4] Socher, R.; Perelygin, A.; Wu, J. Y.; Chuang, J.; Manning,C. D.; Ng, A. Y.; and Potts, C. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In EMNLP.

[5] Evaluation of Clustering. https://nlp.stanford.edu/IR-book/html/htmledition/evaluation-of-clustering-1.html#fig:clustfg3. Accessed 19 Dec. 2020.

[6] Vecto-Ai. (2018). Vecto-ai/word-benchmarks. Retrieved December 04, 2020, from https://github.com/vecto-ai/word-benchmarks

[7] Herzig, Jonathan, et al. “Emotion Detection from Text via Ensemble Classification Using Word Embeddings.” Proceedings of the ACM SIGIR International Conference on Theory of Information Retrieval, ACM, 2017, pp. 269–72. DOI.org (Crossref), doi:10.1145/3121050.3121093.

[8] K. Radinsky, E. Agichtein, E. Gabrilovich, and S. Markovitch. 2011. A word at a time: computing word relatedness using temporal semantic analysis. InWWW ’11, pages 337–346.

[9] Evgeniy Gabrilovich Yehuda Koren Guy Halawi,Gideon Dror. 2012. Large-scale learning of word relatedness with constraints.KDD, pages 1406–1414.

[10] H. Rubenstein and J. Goodenough. 1965. Contextual correlates of synonymy. Communications of the ACM,8:627–633, October.

[11] Thang Luong, Richard Socher, and Christopher Man-ning. 2013. Better word representations with recur-sive neural networks for morphology. In Proceed-ings of CoNLL, pages 104–113, Sofia, Bulgaria.

[12] Camacho-Collados, José & Pilevar, Mohammad Taher & Collier, Nigel & Navigli, Roberto. (2017). SemEval-2017 Task 2: Multilingual and Cross-lingual Semantic Word Similarity. 15-26. 10.18653/v1/S17-2002.

[13] Hill, F., Reichart, R., and Korhonen, A. (2015). Simlex-999: Evaluating semantic models with (genuine) similarity estimation.Computational Linguistics, 41(4):665–695, December

[14] Baker, S., Reichart, R., & Korhonen, A. (2014). An unsupervised model for instance level subcategorization acquisition. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), (pp. 278–289).

[15] L. Finkelstein, E. Gabrilovich, Y. Matias, E. Rivlin,Z. Solan, G. Wolfman, and E. Ruppin. 2001. Placing search in context: The concept revisited. In WWW, pages 406–414. ACM.

[16] E. Agirre, E. Alfonseca, K. Hall, J. Kravalova, M. Pas¸caand A. Soroa. 2009. A study on similarity and relatedness using distributional and wordnet-based approaches. In NAACL '09, pages 19–27.

[17] Mikolov, T., Yih, W., & Zweig, G. (2013). Linguistic Regularities in Continuous Space Word Representations. In HLT-NAACL (pp. 746–751). Retrieved from http://www.aclweb.org/anthology/N13-1#page=784

[18] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. In Proceedings of International Conference on Learning Representations (ICLR).

[19] A. Almuhareb, "Attributes in lexical acquisition," Ph.D. dissertation,University of Essex, 2006.

[20] M. Baroni and A. Lenci, "How we blessed distributional semantic evaluation," inProceedings of the GEMS 2011 Workshop on GEometricalModels of Natural Language Semantics. Association for ComputationalLinguistics, 2011, pp. 1–10.

Appendix A

Similarity/Relatedness

Mturk-287.csv
Recommended by Wang as Generic Evaluation Set: No
Part of Speech: Not specified
Description of Set: Dataset collected from Amazon Mechanical Turk users. 287 pairs assessed by semantic relatedness with a scale from 0 to 5 [8].

Mturk-771.csv
Recommended by Wang as Generic Evaluation Set: Yes
Part of Speech: Not specified Description of Set: Dataset collected from Amazon Mechanical Turk users. 771 pairs assessed by semantic relatedness with a scale from 0 to 5 [9]

rg-65.csv
Recommended by Wang as Generic Evaluation Set: No
Part of Speech: Noun
Description of Set: Classic Rubenstein and Goodenough dataset from 1965 testing the similarity 65 noun pairs with 51 subjects. The subjects score the similarity of the words on a discrete scale from 0 to 4 [10].

Rw.csv
Recommended by Wang as Generic Evaluation Set: No
Part of Speech: Not specified
Description of Set: The Stanford Rare Word (RW) Similarity Dataset with 2 034 pairs of words with low occurrences assessed by semantic similarity with a scale from 0 to 10 [11].

semeval17.csv
Recommended by Wang as Generic Evaluation Set: Yes
Part of Speech: Not specified
Description of Set: 500 pairs assessed by semantic similarity with a scale from 0 to 4 prepared for the SemEval-2017 Task 2 (Multilingual and Cross- lingual Semantic Word Similarity) [12]. Notably, the dataset contains not only words, but also collocations (e.g. climate change)."

simlex999.csv
Recommended by Wang as Generic Evaluation Set: No
Part of Speech: Not specified Description of Set: 999 pairs assessed with a strong respect to semantic similarity with a scale from 0 to 10 [13].

verb-143.csv
Recommended by Wang as Generic Evaluation Set: No
Part of Speech: Verbs
Description of Set: 143 pairs of verbs assessed by semantic similarity with a scale from 0 to 4 [14].

wordsim353-rel.csv
Recommended by Wang as Generic Evaluation Set: Yes
Part of Speech: Not specified
Description of Set: 252 pairs, a subset of WordSim-353 containing pairs that are related (not similar) such as 'family' and 'planning'. Scored with a scale from 0 to 10 [15].

wordsim353-sim.csv
Recommended by Wang as Generic Evaluation Set: Yes
Part of Speech: Not specified
Description of Set: 203 pairs, a subset of WordSim-353 containing semantically similar or unassociated (to mark all pairs that receive a low rating as unassociated) pairs [16].

wordSim353.csv
Recommended by Wang as Generic Evaluation Set: Yes
Part of Speech: Not specified
Description of Set: 353 pairs assessed by semantic similarity (however, some researchers find the instructions for assessors ambiguous with respect to similarity and association) with a scale from 0 to 10 [15].

Analogy

MSR
Analogies dataset of syntactic (i.e. morphological) questions only. Composed of 8,000 questions with 8 kinds of relations. [17]

Google
An unbalanced analogies dataset with 8,869 semantic and 10,675 syntactic questions, with 20-70 pairs per category; *country:capital* relation is over 50% of all semantic questions. Relations in the syntactic part largely the same as MSR. [18]

Concept Categorization

AP
The AP dataset is used for concept categorization and contains 402 words that are divided into 21 categories. [19]

BLESS
The BLESS dataset is used for concept categorization and consists of 200 words divided into 27 semantic classes. [20]