# A1 APPENDIX

Table A1: Table of pre-trained models with unique huggingface identifiers. All models are used through huggingface (Wolf et al., 2019), processed in pytorch (Paszke et al., 2017), and final linear models are fit via scikit-learn (Pedregosa et al., 2011).

| | BERT | DistilBERT |
|---|---|---|
| Base (no finetuning) | `bert-base-uncased` (Devlin et al., 2018) | `distilbert-base-uncased` (Sanh et al., 2019) |
| Emotion | `nateraw/bert-base-uncased-emotion` | `aatmasidha/distilbert-base-uncased-finetuned-emotion` |
| Financial phrasebank | `ahmedrachid/FinancialBERT-Sentiment-Analysis` (Hazourli, 2022) | `yseop/distilbert-base-financial-relation-extraction` (Akl et al., 2021) |
| Rotten tomatoes | `textattack/bert-base-uncased-rotten_tomatoes` (Morris et al., 2020) | `textattack/distilbert-base-uncased-rotten-tomatoes` (Morris et al., 2020) |
| SST2 | `textattack/bert-base-uncased-SST-2` (Morris et al., 2020) | `distilbert-base-uncased-finetuned-sst-2-english` |

| | RoBERTa (Liu et al., 2019) |
|---|---|
| Emotion | `bhadresh-savani/roberta-base-emotion` |
| Financial phrasebank | `abhilash1910/financial_roberta` |
| Rotten tomatoes | `textattack/roberta-base-rotten-tomatoes` (Morris et al., 2020) |
| SST2 | `textattack/roberta-base-SST-2` (Morris et al., 2020) |

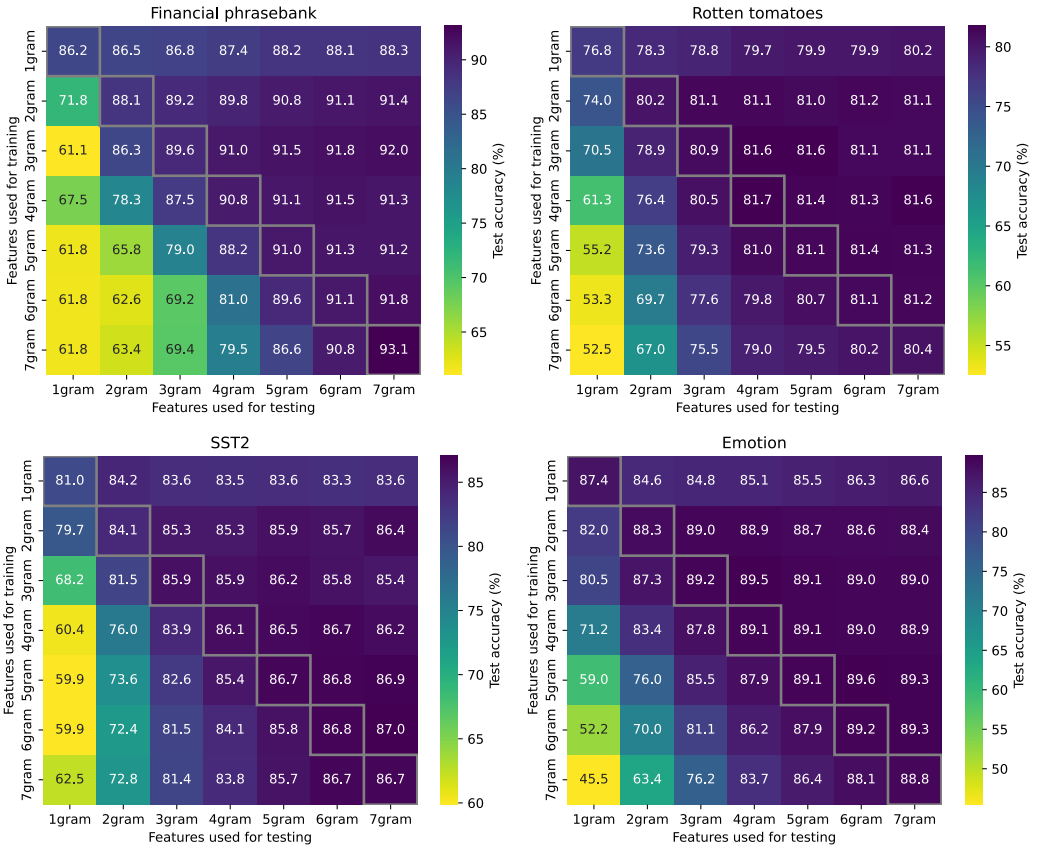


Figure A1: Varying the order of ngrams used for training and testing across each of the five datasets in Table 1. Some models (i.e. rows) perform reasonably well as the order of ngrams used for testing is varied, potentially enabling a test-time tradeoff between accuracy and interpretability. Generally, using higher-order ngrams during testing improves performance and testing with less ngrams than used for training hurts performance considerably.

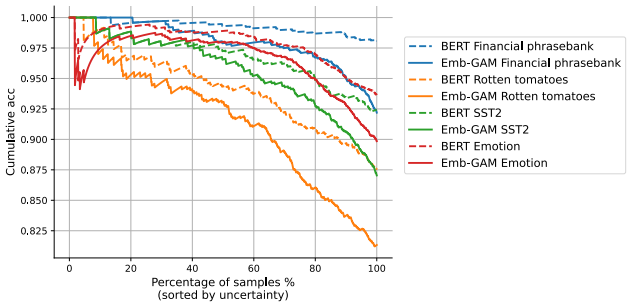


Figure A2: Performance is somewhat calibrated.

By default (Table 3), we use the final embedding layer of the model (and average it over the sequence length to get a fixed size vector), but Table A2 also shows results using the *pooler output* layer of the BERT model. The choice of layer (i.e. final embedding layer versus pooler output) does not seem to make a large difference in the final performance results. Table 3 also shows one variation of the model (*BERT finetuned (noun chunks)*) where rather than training on all ngrams, the model is fit to only noun-phrases extracted by spaCy's dependency parser (Honnibal & Montani, 2017). This results in a performance drop across the datasets, suggesting that these noun-phrases alone are insufficient to perform the classification task.

Table A2: Generalization accuracy varies depending on the model used to extract embeddings. Finetuning the embedding model improves Emb-GAM performance, using a BERT model seems to outperform a DistilBERT model, and the layer used to extract embeddings does not have too large an effect. Top two methods are bolded in each column.

| | Financial phrasebank | Rotten tomatoes | SST2 | Emotion |
|---|---|---|---|---|
| BERT finetuned | **92.8%** ± 0.37% | **81.6%** ± 0.05% | 86.9% ± 0.1% | **89.5%** ± 0.03% |
| BERT finetuned (pooler output) | **93.5%** ± 0.05% | 81.3% ± 0.13% | **87.8%** ± 0.21% | **89.8%** ± 0.07% |
| BERT finetuned (noun chunks) | 87.9% ± 0.08% | 79.7% ± 0.45% | 84.1% ± 0.14% | 87.1% ± 0.2% |
| BERT | 84.1% ± 0.08% | 78.1% ± 0.16% | 82.8% ± 0.27% | 67.1% ± 0.06% |
| BERT (pooler output) | 82.7% ± 0.28% | 78.5% ± 0.03% | 80.7% ± 0.11% | 58.0% ± 0.29% |
| DistilBERT finetuned | 85.8% ± 0.34% | 78.5% ± 0.34% | 81.7% ± 0.07% | 68.8% ± 0.11% |
| DistilBERT | 81.7% ± 0.34% | 79.8% ± 0.08% | 86.8% ± 0.1% | 87.5% ± 0.11% |
| RoBERTa finetuned | 77.8% ± 0.31% | **83.6%** ± 0.03% | **89.1%** ± 0.24% | 88.5% ± 0.19% |

**Summing embeddings meaningfully captures interactions** One potential concern with the Emb-GAM model is that it may fail to learn interactions since it simply sums the embeddings of individual ngrams, and the language model extractor may not sufficiently capture interactions in its embedding space. To investigate this concern, we first identify bigrams that involve interaction by fitting a unigram bag-of-words model and a bigram bag-of-ngrams model to *SST2*. We then use these two models to select the 10 bigrams for which the bigram coefficient is farthest from the sum of the coefficients for each unigram.

Fig A3 shows the resulting bigrams containing interactions. For each bigram, it shows the Emb-GAM learned coefficient (i.e. the contribution to the prediction $w^T \phi(x_i)$) for the bigram (gray bar) along with each of its constituent unigrams (blue and orange bars). It is clear that the bigram

coefficient is not the simple naive sum of the unigram coefficients (dashed black bar), and the learned coefficients make intuitive sense, suggesting that this Emb-GAM model has successfully learned interactions.

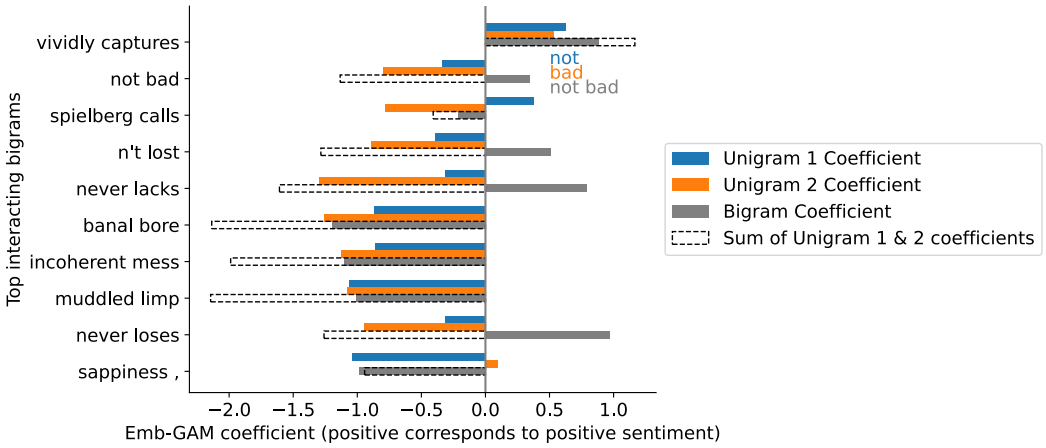


Figure A3: Emb-GAM accurately learns interactions rather than simply summing the contributions of individual unigrams.