Emotion Classification: Explore Structures in Labels

Hao Yan and **Shuchen Yan** and **Tianshu Xin**
University of Wisconsin-Madison

1 Overview of project idea

1.1 Emotion Detection

Emotion detection in Natural Language Processing (NLP) is the task of identifying and categorizing the emotional content of text. It is a subfield of sentiment analysis, which is the process of analyzing opinions, attitudes, and emotions expressed in text data. Emotion detection in NLP has gained increasing attention due to its potential applications in various fields, including marketing, psychology, healthcare, and social media analysis. Emotions play a crucial role in human communication. With the increasing importance of customer feedback in shaping product development and marketing strategies, accurately detecting and analyzing emotions in reviews is becoming a critical task. By detecting and classifying emotions in text data, NLP systems can gain a deeper understanding of the underlying sentiment, leading to a more accurate and nuanced interpretations of human language.

The goal of emotion detection is to automatically classify text into predefined emotion categories such as happiness, sadness, anger, fear, and surprise. This is typically done using machine learning algorithms, which learn to recognize patterns in text data that are associated with different emotions.

Emotion detection involves several stages, including data preprocessing, feature extraction, model training, and evaluation. In the data preprocessing stage, text data is cleaned, normalized, and tokenized into individual words or phrases. In the feature extraction stage, various linguistic features such as word frequency, part-of-speech tags, and sentiment scores are extracted from the text data. These features are then used to train machine learning models, such as support vector machines (SVM), neural networks, or decision trees, to classify text into different emotion categories.

There are several challenges in emotion detection, including the ambiguity and subjectivity of emotion expressions, the lack of labeled data, and the cross-lingual and cross-cultural variations in emotion expression. To address these challenges, researchers have developed various techniques such as multi-lingual and cross-lingual models, transfer learning, and data augmentation.

Overall, emotion detection is an important research area with potential applications in a wide range of fields. With the development of more advanced machine learning techniques and the availability of large annotated datasets, emotion detection in NLP is expected to become increasingly accurate and reliableIt also has a wide range of applications, including building empathetic chatbots (Rashkin et al., 2019), early identification of mental health issues (Greco et al., 2023), and online moderation and cyberbullying detection.

1.2 Project idea

Our project will be developed based on GoEmotions Fine-grained Emotion Detection (Demszky et al., 2020). GoEmotions is an exciting development in NLP that aims to recognize human emotions, which require the developer to employ state-of-the-art deep learning techniques to classify text reviews into twenty-eight fine-grained emotions. Future research can investigate the cross-cultural validity of emotion ratings and expand the taxonomy to encompass other languages and domains.

This task can be formulated as a multi-label classification problem, under which setting the goal is to identify all the emotion labels an input text sequence conveys. One unique aspect of the emotion classification problem is that there are correlations between different emotion labels, also known as label correlation. For example, one would associate the emotions “joy” and “optimism” with positive sentiments, while associate “sadness” and “anger” with negative sentiments. Hence, it is sensible to regard “joy” as being closer to “optimism” than “sadness” or “anger”.

The rest part of this report, which summarizes our work in this project, will develop as follows: In Section 2, we present a short literature survey of the related research in emotion detection and mention some of them which may be considered in our project. In Section 3, we introduce the baseline method in GoEmotion and proposed several ideas beyond the existing work. In Section 4, we describe the dataset to which our project applied, and our results, together with the baseline result, will be shown in Section 5. At the last, we will conclude and discuss about the strengths and limitations of our work, and highlight potential future directions.

2 Literature review

Prior work on emotion classification proposes both feature-based models and neural network models. For example, Mohammad (2018) aims to generate structured, self-attentive, and context-aware sentence embeddings that can be used for a wide range of NLP tasks. As for neural network models, Hsu and Ku (2018) provided more in-depth discussions of the different approaches used by the participating systems, such as recurrent neural networks, convolutional neural networks, and hybrid models. We will focus on reviewing the neural approaches. Many existing approaches (Demszky et al., 2020; Mohammad et al., 2018; Hsu and Ku, 2018) focus on exploring different architectures to handle the problem. In GoEmotions (Demszky et al., 2020), the authors established a robust baseline by fine-tuning a BERT model, but the results indicate that there is significant potential for further improvement. In addition, they also demonstrated the applicability of their trained model across various domains by conducting transfer learning experiments. Although model architecture is important in general, it does not address the label correlation issue discussed in Section 1. And our primary focus is on dealing with this issue.

To develop classification methods tailor-made to the emotion classification task, Huang et al. (2021) addresses multi-label emotion classification by proposing a sequence-to-emotion (Seq2Emo) approach. Seq2Emo uses both token-level (by GloVe embeddings (Pennington et al., 2014)) and contextual pretrained embeddings (by ELMo contextual embeddings (Peters et al., 2018)) to encode words in the input sequences. It implicitly models the correlations among emotions through the decoder’s bi-directional LSTM (BiLSTM) hidden states, which is more suited to multi-label classification than individual predictions. The final hidden state of the BiLSTM encoder is then passed through a linear layer to obtain a fixed-length vector representation of the input text sequence. During the decoding process, an LSTM-based decoder is used to make sequential binary predictions on every candidate emotion. The model feeds a learnable emotion embedding as input at each step of the decoder. This enhances the decoder by explicitly indicating which emotion is being predicted at a step. Additionally, it predicts the presence of all emotions in sequence, rather than using softmax to predict the next plausible emotion from all candidates. This prevents exposure bias and eliminates the need to feed previous predictions as input.The results obtained from experiments on the GoEmotions dataset give a macro-average F1 score of 0.473, yielding an insignificant improvement over the baseline. The LSTM-based decoder approach is questionable, as there are no sequential information to capture across the emotion labels.

Alhuzali and Ananiadou (2021) proposes another approach for multi-label emotion detection task, called SpanEmo. They cast the classification problem as a span-prediction problem. SpanEmo takes into consideration both the input sentence and a label set (i.e., emotion classes) for selecting a span of emotion classes in the label set as the output. The encoder BERT is fed with both the label set and the input sentence as input. It receives two segments as input: the first segment represents the set of all emotion classes, and the second segment corresponds to the input text sequence. By feeding the labels and text jointly to the encoder the encoder can effectively interpolate between emotion classes and all words in the input sentence. Hidden representations are also generated for both words and emotion classes, which enables the model to learn associations between the emotion classes and words in the input sentence. However, it suffers from the same issue as Seq2Emo, as the emotion labels are not a text sequence in nature. Besides, if there are hundreds of labels, it would be unreasonable to concatenate them into one sequence and feed them into the encoder.

There also exist other more general approaches to exploit structures in labels. The literature mainly come from the field of computer vision. Zeng et al. (2023) considers adding a CPCC regularizer in the loss function to preserve hierarchical structures in labels. The hierarchical structures can be represented as a tree. Labels sit on the leaves of the tree, and their parent nodes and ancestor nodes encode the hierarchical information. The authors aim to embed each labels into a continuous vector space. The key idea of CPCC is to require that the distances between pairs of labels in the embeded space are correlated with their distances on the tree. A tree metric of two nodes is defined by the length of their shortest path. The problem of this approach is that the hierarchical tree only capture a very limited structure on the labels. A more natural approach would be a weighted tree, but it is unclear how to obtain the weights.

Contrastive learning is an approach to learn similarities across instances. One first generate positive samples of a query instance, treat all the other instances are as negative samples, and try to embed the query instance and positive samples away from negative samples through minimize the InfoNCE loss. Recently, Hoffmann et al. (2022) proposes the Ranking Info Noise Contrastive Estimation (RINCE) method to capture similarities across different classes. Their approach is to rank the similarity between queries and different sets of positive instances, where the sets of positive instances can come from different related labels to the query. Then, the loss function is the sum of the InfoNCE losses between the query and all the different positive sets, where each InfoNCE loss is associated with a temperature parameter tuned according to its rank. In this way, similar labels are also embeded closely in the embedding space.

3 Methods

3.1 Baseline & Multi-label classification

We reimplement the baseline provided by Google (Demszky et al., 2020). The baseline uses the BERT base model to obtain hidden representations of the text data. On top of the hidden representations, the baseline uses a linear classifier layer with a binary cross-entropy (BCE) loss to handle the multi-label classification task.

The BCE loss in general is not a good loss function for multilabel classification. It treats labels independently. In general, it would be desirable to model the probability of selecting a set of labels jointly, or treat the classification problem as a ranking problem (Zhang and Zhou, 2013). But the BCE loss is convenient to work with, so our focus would be to design method that handles label correlations under this loss.

3.2 CPCC regularizer

Besides the baseline, we considered the CPCC regularizer and applied it. These structures can be represented as a tree, where the labels reside as leaves, while their parent and ancestor nodes encode the hierarchical relationships. The authors' objective is to embed each label in a continuous vector space. The fundamental concept behind CPCC involves ensuring that the distances between label pairs in the embedded space correlate with their distances on the tree. The tree metric measures the distance between two nodes based on the length of their shortest path.

3.3 Label Embedding

We suggest that another way to assess the performance of the trained models is to examine the weight matrix of the linear classifier layer. One may interpret this matrix as an embedding matrix of the labels, as its shape is of $H \times C$, where H is the hidden representation size, and C is the number of classes. We may think of the columns represent vectors of the corresponding labels in an embedding space of H-dimensions. The BERT model projects the text data into this embedding space, and the classification head assigns labels to the input text based on the dot-product similarity between the projection and the label embeddings.

Based on this interpretation, we examine the weight matrix to see whether the model captures a reasonable relation between different labels. We visually inspect the TSNE plot of the weight matrix and perform clustering on the dot-product similarity matrix generated from it. Figure 1 shows the TSNE plot of the Google baseline. We see that it indeed groups emotions under the same Ekman emotion class closer. By performing a spectral clustering procedure on the dot-product similarity matrix, we obtain the result in Table 1. Impressively, this baseline already groups the fine-grained emotions quite accurately. Two emotions grouped incorrectly are "approval" and "neutral", and we will seek a way to resolve it in the future. The TSNE plot and the clustering result given by models with the CPCC regularizer are relatively poor and we include them in the Appendix.

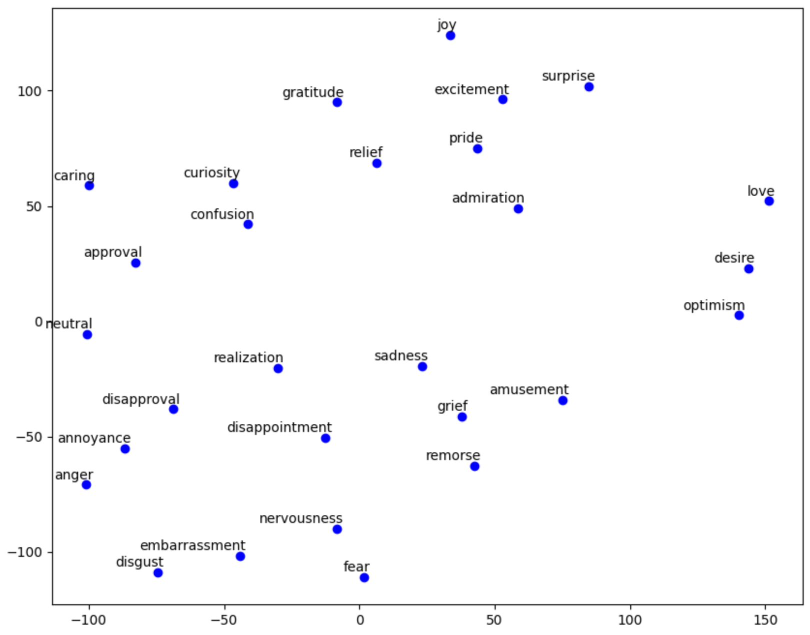

Figure 1: TSNE plot of the weight matrix of the linear classifier layer of the Google baseline.

Cluster	Emotions
1	disgust, fear, nervousness
2	amusement, confusion, curiosity, surprise
3	anger, annoyance, approval, disappointment, disapproval, realization, neutral
4	caring, desire, love, optimism
5	admiration, excitement, gratitude, joy, pride, relief
6	embarrassment, grief, remorse, sadness

Table 1: The clustering result for emotion labels based on the classifier weight matrix of the Google baseline.

3.4 Label similarity

One way to capture the label similarity is to embed the label and encode the similarity into the embeddings. As discussed in Section 3, we can treat the weight matrix in the classifier layer as label embeddings. Then we can require that emotions under the same Ekman class or sentiment group to be close to each other.

We can embed the Ekman classes and the sentiment groups in the same way. One would expect that if an fine-grained emotion embedding provides a good classifier for that emotion, it should also be a reasonable classifier for high level classes such as Ekman classes. Another perspective is that we can imagine a setting in Figure 2. Imagine each point as a fine-grained emotion label, and we would like to separate point B from the other classes. The dotted gray line and dashed gray line each gives a decision boundary with the same misclassification error, but it would be better to choose the dashed line over the dotted line, since it classifies point B and C together rather than A and B. The previous two points are from the same higher level class and are more likely to co-occur, while the latter two points are from different higher level classes. To pick the dashed line over the dotted line, we can compare their angle to the green line, which is the decision boundary for the green higher level class, and pick the one with a smaller angle. This amounts to require the fine-grained label embeddings to be closer to the embeddings of their higher level classes.

Inspired by this observation, we propose a model with an architecture shown in Figure 3. Instead of only working with the original labels, we also include the sentiment labels and the Ekman classes to help the BERT model learn a better hidden repre-

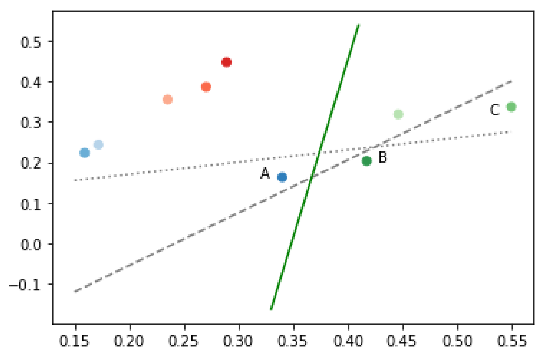

Figure 2: An illustration plot of why the decision boundary of the fine-grained class should be close to the decision boundary of the higher level class. Points denote fine-grained classes, colors indicate higher level classes. The dotted gray line and the dashed gray line denote two decision boundaries for point B. The green solid line represents a decision boundary of the higher class with green color.

sentation. In addition, we add some penalty terms to encourage the fine-grained label embeddings to be close to the embeddings of their corresponding sentiment groups. Denote the embedding of the ith fine-grained label as $\mathbf{w}_{0,i}$, and its corresponding sentiment label and embedding to be $s(i)$ and $\mathbf{w}_{1,s(i)}$, we add a penalty of the form

$$\sum_{i=1}^{C_0} \|\mathbf{w}_{0,i} - \mathbf{w}_{1,s(i)}\|_2^2,$$

where C_0 denotes the number of fine-grained classes. Besides, we also add another penalty term to repel the negative sentiment from the positive sentiment, since it is unlikely for these two sentiments to appear simultaneously in a short Reddit post. Denote the negative sentiment embedding as $\mathbf{w}_{1,neg}$, and the positive embedding as $\mathbf{w}_{1,pos}$, the penalty term we add is

$$-\|\mathbf{w}_{1,neg} - \mathbf{w}_{1,pos}\|_2^2.$$

Combine the penalties together, the training loss is given by

$$\begin{aligned}\sum_{n=1}^{N} \{ &\mathrm{BCE}(\mathbf{W}_0 f(\mathbf{x}_n) + \mathbf{b}_0, \mathbf{y}_n^{(0)}) + \\ &\lambda_1 \mathrm{BCE}(\mathbf{W}_1 f(\mathbf{x}_n) + \mathbf{b}_1, \mathbf{y}_n^{(1)}) + \\ &\lambda_2 \mathrm{BCE}(\mathbf{W}_2 f(\mathbf{x}_n) + \mathbf{b}_2, \mathbf{y}_n^{(2)}) + \\ &\lambda_3 \sum_{i=1}^{C_0} \|\mathbf{w}_{0,i} - \mathbf{w}_{1,s(i)}\|_2^2 - \\ &\lambda_4 \|\mathbf{w}_{1,neg} - \mathbf{w}_{1,pos}\|_2^2 \},\end{aligned} \tag{1}$$

where $\mathbf{x}_n$ is the nth input text, $\mathbf{y}_n^{(0)}$, $\mathbf{y}_n^{(1)}$ and $\mathbf{y}_n^{(2)}$ are the nth fine-grained label, the nth sentiment label, and the nth Ekman label. $f(\cdot)$ is the BERT encoder and λ_i are tuning parameters.

4 Experiment description

4.1 GoEmotions dataset

GoEmotions dataset utilized a data dump from Reddit that originated from the redditdata-tools project, covering comments from the inception of Reddit in 2005 to January 2019 (Demszky et al., 2020). The selection process focused on subreddits with a minimum of 10,000 comments and excluded non-English and deleted comments. It is worth noting that Reddit users predominantly consist of young males, leading to a demographic bias that does not reflect a diverse global population.

The platform has a tendency to feature toxic and offensive language. Several measures have been implemented to curate GoEmotions data (Demszky et al., 2020), including identifying harmful comments through pre-existing lists, which were then used for data filtering and masking. In detail, this selecting and curating process includes: reducing profanity, manual review, sentiment balancing, emotion balancing, subreddit balancing, masking name entity and religion terms.

In creating the taxonomy, the following objectives were considered (Demszky et al., 2020): coverage in terms of emotion expressed in the dataset, coverage in terms of kinds of emotional expression, overlap among emotions, number of emotions.

As for annotation, each example was evaluated by three raters (Demszky et al., 2020). If no agreement on at least one emotion label was reached by the raters, two additional raters were assigned to the example. All raters were native English speakers from India.

Demszky et al. (2020) also provide a detailed analysis of the dataset, including the distribution of emotions, the inter-annotator agreement, and the relationship between emotions and other linguistic features. Besieds, a comparison of the performance of several baseline models for emotion classification on the GoEmotions dataset is also included in that paper.

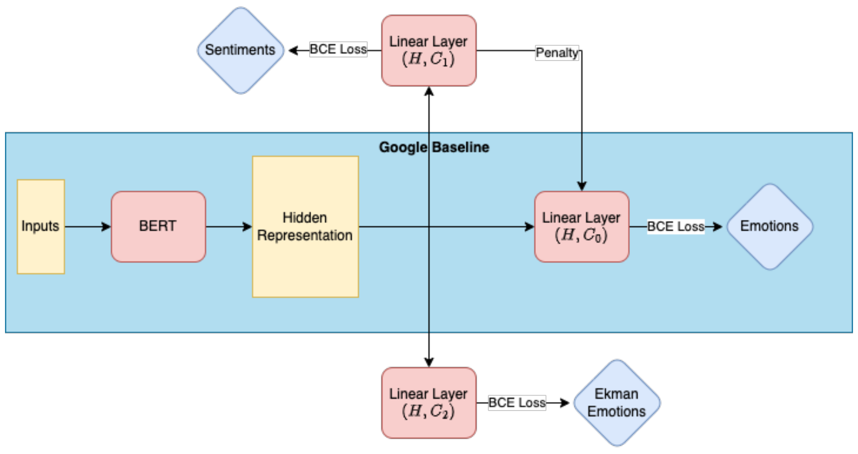

Figure 3: Diagram of the model we propose based on the idea of treating the linear layers as label embeddings and the embeddings of the original fine-grained emotions should be closer to the embeddings of their corresponding sentiment groups. The loss is a weighted sum of the three BCE losses shown in the diagram. C_0 is the number of fine-grained emotion classes, C_1 is the number of sentiment groups, and C_2 is the number of Ekman emotion classes. H is the hidden dimension. The blue box corresponds to Google's Baseline.

5 Results

5.1 Reimplementation

We trained the baseline model for both 10 epochs and 25 epochs and evaluate it on the test dataset using the model with the best macro F1 score on the evaluation dataset. The best result is shown in Table 2. This result is close to Google's result summarized in Table 4 in Demszky et al. (2020). The metric to measure the overall performance is the macro-average F1 score, which is the average F1 scores across all Emotion classes. Notice that the F1 scores of "grief", "pride" and "relief" classes are all 0.

5.2 CPCC regularizer

The tree metric is constructed from the hierarchical structure that the fine-grained emotions each belongs to one of the six Ekman emotion classes, and the latter can be further grouped into four sentiment groups. Following the choice in (Zeng et al., 2023), the tuning parameter λ is set to 0.5 and 1, and the batch size during training is set to 16 and 32. All the other parameters are set the same as Google's baseline. Table 3 summarizes the performance when $\lambda = 0.5$ with a batch size of 32. The model have a higher overall precision. However, it suffers from a lower macro-average F1 score, and it has the same problem of failing at predicting certain classes.

5.3 Label Embedding

We trained the model with the architecture shown in Figure 3 under the loss given in Equation (1) for 25 epochs and evaluate it on the test dataset using the model with the best macro F1 score on the evaluation dataset. We set $\lambda_1 = \lambda_2 = \lambda_4 = 0.05$ and $\lambda_3 = 0.01$ and a batch size of 32. The detail results are summarized in Table 4. We see that a huge advantange over the previous two results is that we did a better job in classifying rarely seen labels such as "grief" "pride" and "relief", and this is crucial in improving the classification precision. Our new model has a macro-average F1 score of 0.494, which also exceeds the baseline score. A TSNE plot of the weight matrix of the linear classifier layer is shown in Figure 4. It can be clearly seen that fine-grained emotion labels are grouped together according to their sentiments.

6 Discussion

6.1 Project summary

We propose a simple model to handle multilabel classification problem, and it out performed the baselines. Our idea is intuitive and may also work well for similar tasks with hierarchical labels.

One issue is that usually, an input text only associates with one or two labels. By using the BCE loss, we do not take this prior information into account. In general, it would be more desirable to estimate a joint distribution over the labels, and classify the labels by maximizing the likelihood function, as this allows we to incorporate more

Emotion	**Precision**	**Recall**	**F1**
admiration	0.590	0.748	0.660
amusement	0.744	0.913	0.820
anger	0.478	0.485	0.481
annoyance	0.362	0.431	0.394
approval	0.439	0.359	0.395
caring	0.435	0.370	0.400
confusion	0.368	0.418	0.391
curiosity	0.451	0.658	0.535
desire	0.567	0.458	0.507
disappointment	0.295	0.344	0.318
disapproval	0.381	0.386	0.384
disgust	0.481	0.504	0.492
embarrassment	0.609	0.378	0.467
excitement	0.462	0.350	0.398
fear	0.598	0.705	0.647
gratitude	0.941	0.909	0.925
grief	0.000	0.000	0.000
joy	0.514	0.689	0.589
love	0.698	0.882	0.779
nervousness	0.500	0.261	0.343
optimism	0.560	0.575	0.568
pride	0.000	0.000	0.000
realization	0.333	0.221	0.266
relief	0.000	0.000	0.000
remorse	0.565	0.857	0.681
sadness	0.464	0.622	0.532
surprise	0.503	0.539	0.521
neutral	0.661	0.655	0.658
macro-average	**0.464**	**0.490**	**0.470**

Table 2: Results based on GoEmotions taxonomy with the reimplemented Google baseline.

Emotion	**Precision**	**Recall**	**F1**
admiration	0.649	0.730	0.687
amusement	0.744	0.913	0.820
anger	0.569	0.439	0.496
annoyance	0.371	0.372	0.371
approval	0.543	0.254	0.346
caring	0.434	0.244	0.313
confusion	0.481	0.333	0.394
curiosity	0.472	0.708	0.566
desire	0.625	0.361	0.458
disappointment	0.362	0.192	0.251
disapproval	0.385	0.375	0.380
disgust	0.630	0.374	0.469
embarrassment	0.632	0.324	0.429
excitement	0.537	0.350	0.424
fear	0.680	0.654	0.667
gratitude	0.944	0.909	0.926
grief	0.000	0.000	0.000
joy	0.629	0.559	0.592
love	0.756	0.845	0.798
nervousness	0.500	0.174	0.258
optimism	0.699	0.538	0.608
pride	0.000	0.000	0.000
realization	0.361	0.090	0.144
relief	0.000	0.000	0.000
remorse	0.612	0.732	0.667
sadness	0.418	0.519	0.463
surprise	0.556	0.489	0.521
neutral	0.625	0.686	0.654
macro-average	**0.508**	**0.434**	**0.454**

Table 3: Results based on GoEmotions taxonomy with the CPCC regularizer. The tuning parameter $\lambda = 0.5$.

prior information. How to come up with a family of suitable joint distribution is challenging, but it is worth investigating in the future.

6.2 Prospective

Another potential direction is treating labels as text. The meaning and difference of emotions can be well captured by their definitions in dictionary. For example, the definition of the emotion "grief" is "Intense sorrow, especially caused by someone's death". The definition provides a much clearer criterion to associate a given text to the emotion "grief". And although "grief" is less commonly seen in the text data, as it is a very extreme emotion, the text data associated "grief" will provide strong signals and high confidence for future text classification once you know the meaning of "grief". Hence, utilizing the definitions will well mitigate the unbalanced class problem, and potentially helps us achieve a much better result. The dictionary definitions also encode similarity and levels of intensity of different labels, providing rich information to be extracted. This idea is also universally applicable when the labels of a classification problem can be well understood by their dictionary definitions. This looks interesting, as it has more potential to address the unbalanced data problem and could be easily extended to other settings. The issue is that this might not be achievable with a small model, as its language understanding capacity is limited. One possibility is to do some prefix-tuning on a large language model, and measure its performance.

It remains a challenge to us of how to encode the definitions. Existing work such as Alhuzali and

Emotion	**Precision**	**Recall**	**F1**
admiration	0.649	0.714	0.680
amusement	0.749	0.894	0.815
anger	0.480	0.495	0.488
annoyance	0.349	0.319	0.333
approval	0.360	0.342	0.351
caring	0.383	0.363	0.373
confusion	0.322	0.451	0.376
curiosity	0.481	0.535	0.507
desire	0.517	0.373	0.434
disappointment	0.280	0.265	0.272
disapproval	0.378	0.382	0.380
disgust	0.510	0.407	0.452
embarrassment	0.615	0.432	0.508
excitement	0.427	0.427	0.427
fear	0.607	0.692	0.647
gratitude	0.935	0.901	0.918
grief	0.333	0.167	0.222
joy	0.601	0.646	0.623
love	0.737	0.824	0.778
nervousness	0.348	0.348	0.348
optimism	0.619	0.532	0.572
pride	0.833	0.312	0.455
realization	0.169	0.166	0.167
relief	1.000	0.182	0.308
remorse	0.550	0.786	0.647
sadness	0.543	0.526	0.534
surprise	0.554	0.617	0.584
neutral	0.634	0.622	0.628
macro-average	**0.534**	**0.490**	**0.494**

Table 4: Results produced by our proposed model.

Ananiadou (2021) and Huang et al. (2021) provide ways to jointly embed labels and text, but their approaches still handle labels different from text. Recent work based on prompt engineering (Liu et al., 2021) used cloze prompts to handle text classification problems. Their focus is more on large language models and few shot learning setting. We would like to have a better understanding of current literature and further explore on this idea.

6.3 Teamwork

We appreciate every team member for their contribution, and also Dr. Junjie Hu, our instructor, for his lectures and guidance. Each of the team member has brought a unique perspective and skill to our project. It was Hao that usually brought us with new promising perspectives from his comprehensive understanding of neural network and emotion detection, and undertook the main part of implementation of the ideas by his proficient engineering skill. Shuchen spent more time in reviewing literature, searching new ideas, organizing project progress, and preparing project reports and presentation. Tianshu engaged in team discussion and took part in writing reports. Undoubtedly, this is a meaningful project which makes our team strengthen our understanding and application of knowledge learned in this NLP class.

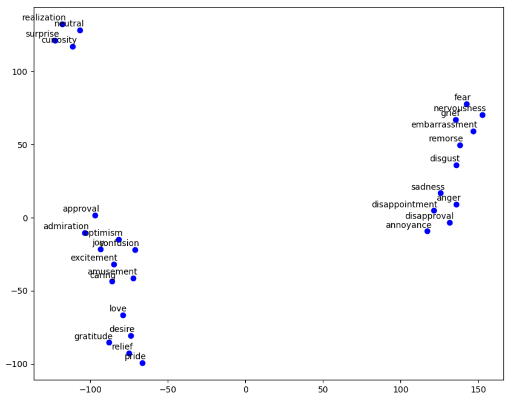

Figure 4: TSNE plot of the weight matrix of the linear classifier layer of our proposed model.

References

Hassan Alhuzali and Sophia Ananiadou. 2021. SpanEmo: Casting multi-label emotion classification as span-prediction. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1573–1584, Online. Association for Computational Linguistics.

Dorottya Demszky, Dana Movshovitz-Attias, Jeongwoo Ko, Alan Cowen, Gaurav Nemade, and Sujith Ravi. 2020. GoEmotions: A Dataset of Fine-Grained Emotions. In *58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Candida M Greco, Andrea Simeri, Andrea Tagarelli, and Ester Zumpano. 2023. Transformer-based language models for mental health issues: A survey. *Pattern Recognition Letters*, 167:204–211.

David T. Hoffmann, Nadine Behrmann, Juergen Gall, Thomas Brox, and Mehdi Noroozi. 2022. Ranking info noise contrastive estimation: Boosting contrastive learning via ranked positives.

Chao-Chun Hsu and Lun-Wei Ku. 2018. SocialNLP 2018 EmotionX challenge overview: Recognizing emotions in dialogues. In *Proceedings of the Sixth International Workshop on Natural Language Processing for Social Media*, pages 27–31, Melbourne, Australia. Association for Computational Linguistics.

Chenyang Huang, Amine Trabelsi, Xuebin Qin, Nawshad Farruque, Lili Mou, and Osmar R Zaiane. 2021. Seq2emo: A sequence to multi-label emotion classification model. In *North American Chapter of the Association for Computational Linguistics*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing.

Saif Mohammad. 2018. Obtaining reliable human ratings of valence, arousal, and dominance for 20,000 English words. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 174–184, Melbourne, Australia. Association for Computational Linguistics.

Saif Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. 2018. SemEval-2018 task 1: Affect in tweets. In *Proceedings of the 12th International Workshop on Semantic Evaluation*, pages 1–17, New Orleans, Louisiana. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. 2019. Towards empathetic open-domain conversation models: A new benchmark and dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5370–5381, Florence, Italy. Association for Computational Linguistics.

Siqi Zeng, Remi Tachet des Combes, and Han Zhao. 2023. Learning structured representations by embedding class hierarchy. In *The Eleventh International Conference on Learning Representations*.

Min-Ling Zhang and Zhi-Hua Zhou. 2013. A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering*, 26(8):1819–1837.