Label-Aware Hyperbolic Embeddings for Fine-grained Emotion Classification

Chih-Yao Chen[1], Tun-Min Hung[2], Yi-Li Hsu[2], Lun-Wei Ku[2]
[1]UNC Chapel Hill, [2]Institute of Information Science, Academia Sinica
[1]cychen@cs.unc.edu
[2]{allenhung,yili.hsu,lwku}@iis.sinica.edu.tw

Abstract

Fine-grained emotion classification (FEC) is a challenging task. Specifically, FEC needs to handle subtle nuance between labels, which can be complex and confusing. Most existing models only address text classification problem in the euclidean space, which we believe may not be the optimal solution as labels of close semantic (e.g., *afraid* and *terrified*) may not be differentiated in such space, which harms the performance. In this paper, we propose HypEmo, a novel framework that can integrate hyperbolic embeddings to improve the FEC task. First, we learn label embeddings in the hyperbolic space to better capture their hierarchical structure, and then our model projects contextualized representations to the hyperbolic space to compute the distance between samples and labels. Experimental results show that incorporating such distance to weight cross entropy loss substantially improves the performance with significantly higher efficiency. We evaluate our proposed model on two benchmark datasets and found 4.8% relative improvement compared to the previous state of the art with 43.2% fewer parameters and 76.9% less training time. Code is available at https://github.com/dinobby/HypEmo.

1 Introduction

Fine-grained classification is a challenging yet important task that involves differentiating subtle distinctions in a label set. For instance, in image classification, classifying cars, planes, and other vehicles is *coarse-grained* classification, whereas distinguishing models from cars is *fine-grained* classification. In NLP, sentiment analysis, which attempts to classify positive/negative sentiments, is an example of coarse-grained text classification. Human emotions, however, exhibit more complexity. For example, the six type of basic emotion (Ekman, 1999) include *happiness, sadness, fear, disgust, anger*, and *surprise*, and show finer distinctions of positive and negative classes. Moreover,

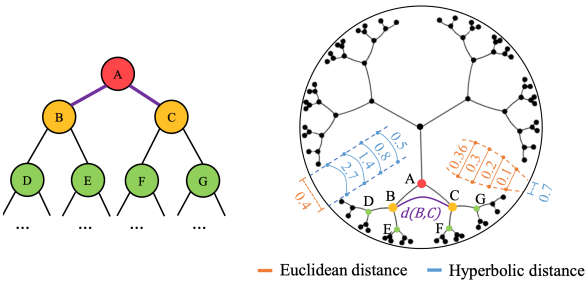


Figure 1: Properties of hyperbolic space. The hyperbolic distance between two points grows dramatically as they come close to the border, converse to the Euclidean distance. Moreover, the hyperbolic distance is an analog of tree distance, which is $d(B, C) \approx d(B, A) + d(A, C)$.

complex interactions exists in human emotion as different type of emotions can have subtle differences, for instance *ashamed* and *guilty*. This makes fine-grained emotion classification (FEC) challenging not only because of the increased number of classes, but also because of the increased similarity between classes. For instance, the current finest emotion classification datasets contain up to 27 and 32 classes of emotion (Rashkin et al., 2019; Demszky et al., 2020), respectively. Classes such as *furious* and *angry* in these fine-grained datasets are far more difficult to differentiate than *happy* and *sad*. However, detecting fine-grained emotion is useful in a variety of situations. For example, in a dialog generation system, understanding the user's fine-grained emotion could facilitate more empathy in the responses, which might increase conversation engagement (Roller et al., 2021). Emotions also play an important role in people's lives, affecting how they make decisions and how they interact with machines. Therefore, the finer we can classify, the more information we can collect to model users' behaviors.

Existing text classification methods often use pre-trained language models such as BERT (Devlin et al., 2019) to generate a contextualized represen-

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 10947–10958
July 9-14, 2023

tation, and fine-tune them primarily in Euclidean space for downstream tasks. However, such a space is limited, as some confusing pairs have nearly identical meanings (e.g., *furious* and *angry*), and forcing them to be separated in the latent space may harm the performance by overfitting on training data. The complex nature of emotions can be expressed in a hierarchical way, consisting of three levels (Parrott, 2001). For instance, *Joy*, *Contentment*, and *Pleasure* are primary, secondary, and tertiary emotions, respectively. Meanwhile, learning embeddings in hyperbolic space is becoming more popular due to its superior ability to capture hierarchical information (Nickel and Kiela, 2017; Ganea et al., 2018; Chami et al., 2019; Liu et al., 2019a). Figure 1 demonstrates a tree embedded to hyperbolic space. Tree nodes closer to the root are embedded near the origin, and nodes closer to leaves are placed closer to the boundary. The main merit of this space is that as the distance from the origin rises, the amount of space in hyperbolic space grows exponentially (Cho et al., 2019; López and Strube, 2020; Peng et al., 2021). Intuitively, tree-like structures also expand the number of nodes as the distance increases from the root, which is consistent with the mathematical basis of hyperbolic geometry. This is also reflected in the hyperbolic distance (Eq. 6), which resembles the distance between two nodes in a tree.

In this work, we propose HypEmo, which integrates label embedding trained in hyperbolic space with a RoBERTa model fine-tuned in Euclidean space. Specifically, we first learn hierarchical-aware label embeddings in hyperbolic space, and then project the representation output by RoBERTa onto the same space to derive the distance between a text representation and its corresponding label. This distance is then used to weight standard cross entropy loss, making the projection of text representation as close to its label as possible, in hyperbolic space. Results on two challenging datasets, GoEmotions (Demszky et al., 2020) and EmpatheticDialogs (Rashkin et al., 2019), demonstrate the superiority of the proposed model. Also, we find that HypEmo performs best when the label structure is more complex, or the inter-class relationship is more ambiguous. To sum up, the contributions of this paper are threefold:

- We leverage the merits of hyperbolic geometry to learn better representations in both hyperbolic and Euclidean space.
- We propose the novel HypEmo framework along with a simple yet effective objective function to address the FEC task.
- Empirically, the proposed model outperforms existing methods, and is even comparable with systems that utilize external knowledge or data augmentation techniques.

2 Related Work

2.1 Fine-grained Classification

Most of the literature addresses fine-grained text classification in Euclidean space. Khanpour and Caragea (2018) propose combining lexicon-based features to detect fine-grained emotions in online health posts. Yin et al. (2020) demonstrate that pre-trained models can learn compositional sentiment semantics with self-attention applied to a binary constituency parse tree and transfer to downstream sentiment analysis tasks. Mekala et al. (2021) propose utilizing generative language models for fine-grained classification on coarsely annotated data. Suresh and Ong (2021) propose label-aware contrastive loss (LCL), which estimates the model confidence for each sample, and use this to weight supervised contrastive loss (Khosla et al., 2020). All of the above-mentioned work addresses FEC task primarily on the euclidean space, while we argue that some emotion with close semantics are not separable in this latent space. In our work, we integrate hyperbolic space to address this issue that improves FEC task.

2.2 Hyperbolic Geometry

A hyperbolic space is a non-Euclidean space for which the parallel postulate does not hold (Peng et al., 2021; Dhingra et al., 2018). The parallel postulate asserts that for every line L and point P not on L, there is a unique line that passes through P that shares the same plane with L and P and yet does not intersect with L. Without this postulate, familiar mathematical properties in Euclidean space are different. For example, in hyperbolic space, there can be more than one line parallel to line L that goes through a point P not on L. Also, whereas the distance between two points is a straight line in Euclidean space, this can be generalized as a geodesic $\in [0, 1]$ which is the minimized distance between two points. Moreover, the hyperbolic distance grows exponentially as the points approach the boundary, making it more spacious

than Euclidean space given the same dimensions. These properties suit the nature of a tree-like structure, as the number of nodes grows exponentially when the depth increases. In our work, we leverage the nature of hyperbolic geometry to better capture label hierarchy, and propose improving the FEC task by jointly learning representations in both Euclidean and hyperbolic space.

2.3 Poincaré Embeddings

The Poincaré ball model (Cannon et al., 1997) is commonly adopted in hyperbolic neural networks (HNN) and representation learning research due to its differentiable distance function (Dhingra et al., 2018; Nickel and Kiela, 2017). The Poincaré ball model is a Riemannian manifold that can be formulated as (B, g_x^b) with a Riemannian metric $g_x^b = \lambda_x^2 g^E$, where $\lambda_x^2 = \frac{2}{1-\|x\|^2}$ is called the *conformal factor*, and $g^E = I_n$ is the Euclidean metric tensor. The Riemannian metric defines the geometric properties of a space, such as distances, angles, or curve length. For example, Euclidean space is a manifold with zero curvature, and the distance between two points can be written as $d(x, y) = \sqrt{\sum_i (x_i - y_i)^2}$. The Poincaré ball model, on the other hand, is a manifold with a constant negative curvature, where $B = \{x \in \mathbb{R}^n : \|x\| < 1\}$ is an unit ball. In natural language processing, researchers have applied such embeddings to tasks as varied as fine-grained entity typing (López and Strube, 2020), text classification (Cho et al., 2019), and language modeling (Dhingra et al., 2018).

3 Methodology

In this section, we describe the proposed HypEmo in detail. Fig. 2 illustrates its workflow.

3.1 Euclidean Sequence Representations

Given the input sequence $x_i = \{x_i^1, x_i^2, \ldots, x_i^k\}$ with k tokens, text encoder generates the representation of the input sequence h_i. The text encoder itself is model agnostic, which could be any transformer-like model or even trained directly on the hyperbolic space. Here we use $\text{RoBERTa}_{\text{base}}$ in particular, since its excellence of generating contextualized representations. We also discuss the generalization of BERT-style models as the text encoder in the experiment. As convention, we take the hidden states corresponding to [CLS] token as the sequence representation h_i.

3.2 Hyperbolic Projection

We adopt the Poincaré ball model of hyperbolic space. Let B be the Poincaré ball model, and the associated tangent space denoted as $\tau_x B$, we use exponential map $exp_x : \tau_x B \to B, \forall x \in B$, to project points from euclidean space to the hyperbolic space:

$$\exp_x(v) = x \oplus \tanh(\frac{\lambda_x \|v\|}{2}) \frac{v}{\|v\|} \quad (1)$$

On the contrary, we could use logarithmic map to project points back to the euclidean space if needed:

$$\log_x(y) = \frac{2}{\lambda_x} \tanh^{-1}(\| -x \oplus y\|) \frac{-x \oplus y}{\| -x \oplus y\|} \quad (2)$$

where $v \neq 0$ and $y \neq x$ is the tangent vector, $\lambda_x = \frac{2}{1-\|x\|^2}$ is the conformal factor, and $\oplus$ is the Möbius addition:

$$x \oplus y = \frac{(1 + 2\langle x, y\rangle + \|y\|^2)x + (1 - \|x\|^2)y}{1 + 2\langle x, y\rangle + \|x\|^2\|y\|^2} \quad (3)$$

With exponential and logarithmic map, we can project embeddings from euclidean space to hyperbolic space or vice versa, and hence allowing us to take advantage on both spaces. Specifically, most of the well known language models are pretrained on euclidean space which are powerful and easy to use, while in the hyperbolic space we can better model the label inventories in order to boost the performance.

3.3 Hyperbolic Label Embeddings

To fully utilize the hierarchy of label set, we train label representations on the hyperbolic space. In this stage, our goal is to learn representations for each class $\mathcal{E} = \{e_1, e_2, \ldots, e_m\}$, where e_i is the hyperbolic label embeddings $\in \mathbb{R}^{h_d}$, h_d is the dimension of hyperbolic space, and m is the number of classes. The label set can be represented as a set of tuples, indicating the parent-children relationship between nodes: $\mathcal{D} = \{(u, v)\}$ where u is the parent of v. For datasets which does not contain parent-children relationship, we follow the parrot's emotion model (Parrott, 2001) to form $\mathcal{D}$, which has at most three levels of hierarchy. For the objective, we follow previous work (Nickel and Kiela, 2017; Ganea et al., 2018) that maximizes the distance between unrelated samples using negative

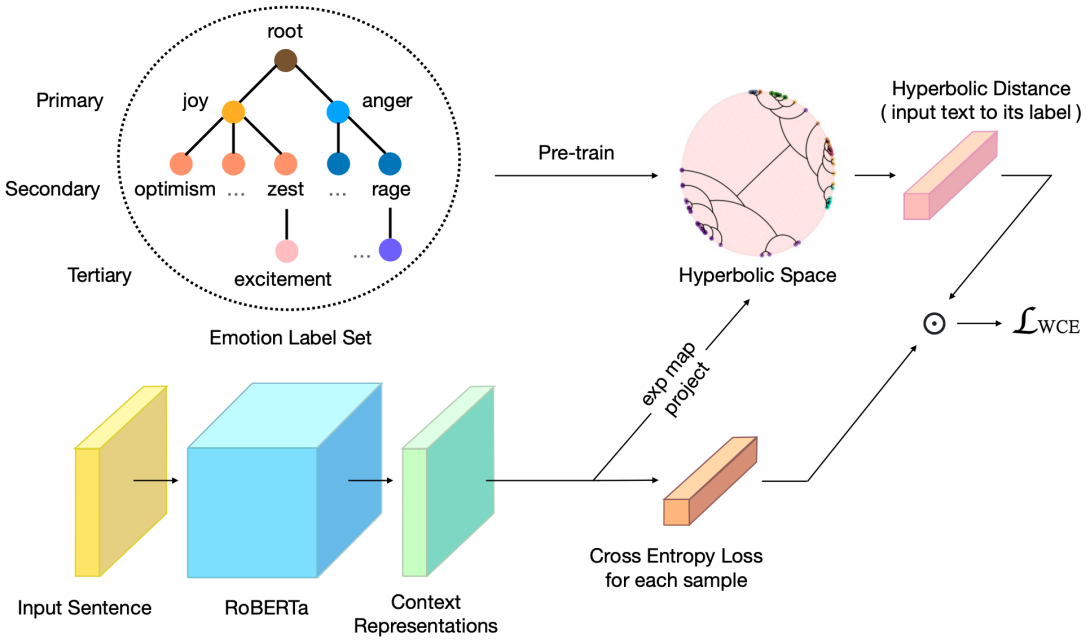


Figure 2: The overall architecture of proposed framework. Label embeddings are pre-trained on the hyperbolic space to capture the hierarchical information, and the input sequence is encoded by RoBERTa to generate contextualized representations h_i and cross entropy loss for each sample. h_i will be simultaneously projected to the hyperbolic space to derive the hyperbolic distance w_i to its label, and w_i will be used to weight the standard cross entropy loss.

sampling:

$$\mathcal{L}_{label} = -\sum_{(u,v)\in\mathcal{D}} log\frac{e^{-d(u,v)}}{\sum_{v'\in\mathcal{N}(u)\cup\{v\}} e^{-d(u,v')}} \tag{4}$$

where $\mathcal{N}(u) = \{v : (u,v) \notin \mathcal{D}, v \neq u\}$ is the negative sample set and $d(u,v)$ is the distance between two points calculated by Eq. 6. We use Riemammian Adam (Becigneul and Ganea, 2019) for the optimization. After that, hyperbolic embeddings for each label is ready for use in the next step.

3.4 Label-Aware Hyperbolic Weighted Loss

Cross entropy loss is commonly used in classification task. It assumes that every instance's negative log probability contributes equally. Usually, classifying a sample to be *furious* when the ground truth is *angry*, is more forgivable than classifying a sample to be *joy*. However, it is the subtle difference between those confusable pairs such as *angry* and *furious* that makes fine-grained classification task challenging. In our work, we incorporate hyperbolic distance to enhance learning efficacy. To be more specific, we expect the confusable pairs that shares almost identical semantics can be well seperated on the hyperbolic space, and jointly update the model lies in the euclidean space. Formally, the pre-trained hyperbolic label embedding set (See Sec.3.3) is denoted as $\mathcal{E} = \{e_1, e_2, \ldots, e_m\}$. Each instance $\{x_i, y_i\}$ contains a pair of sequences and labels where $y_i \in \mathcal{M}$ and $\mathcal{M}$ denotes the label set with $|\mathcal{M}| = m$. Given the sequence x_i, h_i is extracted from the text encoder, and the logit $c_i \in \mathbb{R}^m$ is obtained by further passing h_i through a linear layer: $c_i = \text{MLP}(h_i)$. The standard cross-entropy loss is expressed mathematically as follows.

$$\mathcal{L}_{\mathcal{CE}} = \frac{1}{N}\sum_{i=1}^{N} -log\frac{exp(c_i^{y_i})}{\sum_{j=1}^{K} exp(c_i^j)} \tag{5}$$

The length of the geodesic, i.e., the distance between two points in a Poincaré ball is given by:

$$d(x_i, y_i) = \cosh^{-1}(1+2\frac{\|x_i - y_i\|^2}{(1-\|x_i\|^2)(1-\|y_i\|^2)}) \tag{6}$$

Now we can project the text representation generated from the encoder to the hyperbolic space using Eq. 1, and calculate the distance $w \in \mathbb{R}$ of text representation and the label embeddings, which are both on the hyperbolic space. We expect the embeddings for both input sequence and its label to be as close as possible, which means the distance w is expected to be minimized. A simple way is to integrate w into $\mathcal{L}_{\mathcal{CE}}$ by multiplying:

$$\mathcal{L}_{\mathcal{WCE}} = \sum_{i=1}^{N} -w_i log\frac{exp(c_i^{y_i})}{\sum_{j=1}^{K} exp(c_i^j)} \tag{7}$$

In this way, w can be viewed as a weight to penalize pairs that have larger distance in the hyperbolic space. Our goal here is to jointly learn embeddings in both Euclidean and hyperbolic space to boost the performance. By taking w as a weight to sum up cross entropy loss, the main merit is that it is easy to implement without carrying all optimization operation on the hyperbolic space, while allowing the whole framework to be updated jointly. We will also discuss the scenario when fully using hyperbolic neural networks in Sec. 4.3.

4 Experiment

4.1 Datasets

We evaluate our method on two datasets: GoEmotions (Demszky et al., 2020), and Empathetic Dialogues (Rashkin et al., 2019). Given our primary objective of fine-grained emotion classification, which requires distinguishing labels with subtle differences in meaning, we choose GoEmotions and Empathetic Dialogues for evaluation. These datasets are considered the most challenging datasets, as they contain a larger number of emotion labels with similar semantics. Below we give the descriptions of these datasets. GoEmotions is composed of comments from Reddit (Demszky et al., 2020). The total number of samples is 54k, and each sample is annotated with one or multiple labels among 27 emotions and neutral. To ensure a fair comparison with previous work (Suresh and Ong, 2021), we use only single-labeled samples and exclude neutral ones. The training/validation/test split of the remaining dataset is 23,485 / 2,956 / 2,984.

Empathetic Dialogues (Rashkin et al., 2019) consists of conversations with single emotion labels. The situation was written by a speaker given an emotion label. The listener was to respond with an empathetic utterance. The process could go on for up to six turns. Since the situation was written based on the provided emotion, we used the situation as the model's input, following Suresh and Ong (2021). The dataset contains 24,850 conversations labeled among 32 emotions. The training/validation/test split of the dataset is 19,533 / 2,770 / 2,547, respectively. Below, we use GE to represent GoEmotions and ED to represent Empathetic Dialogues.

4.2 Experiment Settings and Baselines

We compare the proposed HypEmo primarily with three categories of strong baselines:

General pre-trained language models. We compared against BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019b), and ELECTRA (Clark et al., 2020). These models are pre-trained on large unlabeled corpora. They generate high-quality representations, and all perform well on text classification tasks. We also compare different size of these pre-trained models, denoted as base and large.

Label Embedding-aware models. Suresh and Ong (2021) propose label-aware contrastive loss (LCL), which weights each negative sample differently. Specifically, more confusable pairs contribute more to the objective function; this yields promising results on fine-grained text classification. In addition, we compare against HiAGM (Jie Zhou, 2020), the strongest hierarchy-aware text classification model with source code publicly available. Lastly, we implement a baseline called LabelEmb, which encodes the label description (i.e., the definition of emotions) to derive label embeddings, and train the model on the Euclidean space with the rest setting same as HypEmo.

Hyperbolic classification models. We also compared with models trained in hyperbolic space for classification, including (1) Hyperbolic SVM (HSVM) proposed by Cho et al. (2019), which generalizes the support vector machine to hyperbolic space, and (2) Hyperbolic Neural Model (HNN) proposed by Ganea et al. (2018), a hyperbolic GRU that performs all necessary operations in hyperbolic space to train a neural network. (3) Hyperbolic Interaction Model (HyperIM) (Chen et al., 2020) jointly learns word and label embeddings, and measure the similarities in the Poincoré disk to aggregate input representations. (4) HIDDEN (Chatterjee et al., 2021) is a framework which does not assume the label hierarchy is known. It also proposed to learn the label embedding jointly in an end-to-end fashion. For a fair comparison, we set the word dimension of the hyperbolic space to 100, the same as the dimension we use.

Evaluation metrics. Following Suresh and Ong (2021), we use accuracy and weighted F1 as the evaluation metrics. Weighted F1 takes into account the number of samples in each class, and weights the macro F1 by this ratio. This can be expressed

as

$$F1_{weighted} = 2\sum_{c} \frac{n_c}{N} \frac{P_c \times R_c}{P_c + R_c}, \quad (8)$$

where n_c is the number of samples in class c, N is the number of total samples, and P_c and R_c are the precision and recall for class c, respectively.

Implementation Details. HypEmo is a encoder-agnostic framework which can easily adapt to different kinds of text encoders. In the experiment, we use pre-trained $\text{RoBERTa}_{\text{base}}$ as the backbone, which has 12 layers with a hidden size of 768. During training, we applied the Adam optimizer in Euclidean space with a learning rate of 10^{-5} and a weight decay of 0.01. By contrast, we utilized Riemannian Adam (Becigneul and Ganea, 2019) to train our label embeddings in hyperbolic space with a learning rate of 0.01. The dimension of hyperbolic label embedding is set to 100, which is searched from {2, 10, 50, 100, 250}. Other implementation details can be found in our code.

4.3 Main Results

Baseline comparison. To demonstrate the effectiveness of HypEmo, we conduct experiments to compare the performance of different models. The comparison is shown in Table 1. Firstly, we compare HypEmo with general pre-trained language models. Among them, $\text{RoBERTa}_{\text{large}}$ performs the best, while HypEmo outperforms it on weighted F1 by 2.8% on ED and 1.1% on GE. It is also worth mentioning that HypEmo has considerably smaller parameter size compared with $\text{RoBERTa}_{\text{large}}$ (125M v.s. 355M), resulting in significantly lower training and inference time. This indicates the effectiveness of the proposed label-aware hyperbolic embeddings and the strategy to weight the standard cross entropy loss by hyperbolic distance.

Secondly, we compare with label-aware system. Since LCL is the previous state-of-the-art, we mainly compare the efficiency with it (Δ in the left of Table 1). Results show that our proposed method outperforms LCL with much higher efficiency. This is because LCL augments data by using the synonym replacement technique, which doubles the size of data. Also, they use two encoders to train the main classifier and a weighting network, which doubles the parameter size. In contrast, HypEmo uses single encoder and uses only the original samples without any data augmentation method, and still out-wins LCL by 2.8% and 2.5% absolute F1 score on ED and GE, respectively. Moreover, Although HiAGM take into account the label hierarchy, it utilize RNN architecture, making it less efficient and underperforming HypEmo by a large margin. Lastly, HypEmo performs better than LabelEmb, which LabelEmb calculates the weighted loss in the Euclidean space. This again demonstrates the efficacy of our proposed hyperbolic space integration.

Also, we notice that HypEmo works better than models that are fully trained on hyperbolic space, which indicates the benefits of jointly learning the hyperbolic label embedding and fine-tuning $\text{RoBERTa}_{\text{base}}$ in a hybrid space settings. This hybrid setting could benefit from both the power of pre-trained language model and the strength of hyperbolic space to capture hierarchical information. To sum up, we could achieve better results compared to previous works without increasing data size or model parameters, which is more effective and efficient.

Performance on different encoder. We apply HypEmo on top of different encoders to examine whether HypEmo is a model-agnostic method that could bring improvement regardless of the encoder being used. Table 2 shows the results in terms of weighted F1 score. We observe that no matter which encoder is adopted, adding HypEmo leads to further improvements. For instance, applying HypEmo on $\text{BERT}_{\text{base}}$ enhances the performance by 5.9% absolute percentage on ED, and the same phenomenon can be observed on $\text{RoBERTa}_{\text{base}}$ and $\text{ELECTRA}_{\text{base}}$ across two datasets. This verifies that HypEmo is model-agnostic and could be easily built on top of any text encoder to boost performance.

4.4 Case Study

Following Suresh and Ong (2021), we investigate the performance of HypEmo when the label set contains pairs with close semantics. We compare the proposed HypEmo with different objectives and encoders, and follow the ED subsets selected by Suresh and Ong (2021), which include the most difficult sets chosen after enumerating all combinations that contain four labels. These subsets are a: {*Anxious, Apprehensive, Afraid, Terrified*}, b: {*Devastated, Nostalgic, Sad, Sentimental*}, c: {*Angry, Ashamed, Furious, Guilty*}, and d: {*Anticipating, Excited, Hopeful, Guilty*} from the ED dataset. We conducted the experiments with RoBERTA and

	#Params (↓)	ξ_{tr} (↓)	ξ_{conv} (↓)	ξ_{inf} (↓)	Empathetic Dialogues		GoEmotions	
					Acc (↑)	**F1** (↑)	**Acc** (↑)	**F1** (↑)
$\text{BERT}_{\text{base}}$	110M	96.7	290.0	1.1	50.4 ±0.3	51.8 ±0.1	60.9 ±0.4	62.9 ±0.5
$\text{RoBERTa}_{\text{base}}$	125M	99.3	297.9	1.2	54.5 ±0.7	56.0 ±0.4	62.6 ±0.6	64.0 ±0.2
$\text{ELECTRA}_{\text{base}}$	110M	97.6	292.7	1.1	47.7 ±1.2	49.6 ±1.0	59.5 ±0.4	61.6 ±0.6
$\text{BERT}_{\text{large}}$	340M	181.0	362.0	3.5	53.8 ±0.1	54.3 ±0.1	64.5 ±0.3	65.2 ±0.4
$\text{RoBERTa}_{\text{large}}$	355M	185.7	371.4	3.7	57.4 ±0.5	58.2 ±0.3	64.6 ±0.3	65.2 ±0.2
$\text{ELECTRA}_{\text{large}}$	335M	179.9	539.8	3.5	56.7 ±0.6	57.6 ±0.6	63.5 ±0.3	64.1 ±0.4
LCL†	220M	421.7	1264.9	3.9	59.1 ±0.4	58.2 ±0.5	64.6 ±0.2	63.8 ±0.3
HiAGM	15M	2673.2	10692.9	8.6	47.8 ±0.6	50.2 ±0.7	59.7 ±0.6	61.8 ±0.5
LabelEmb	125M	103.6	518.1	1.1	55.1 ±0.7	56.2 ±0.5	62.7 ±0.6	62.8 ±0.4
HSVM	428K	14.7	42.1	0.7	27.4 ±0.0	26.7 ±0.0	23.6 ±0.0	22.3 ±0.0
HNN	5M	15421.5	92529.1	17.5	41.2 ±0.9	42.0 ±0.8	46.6 ±0.6	47.2 ±0.5
HyperIM	5M	2266.3	6798.9	8.4	44.1 ±1.2	43.6 ±1.0	50.2 ±0.9	49.7 ±0.7
HIDDEN	11M	13473.0	67364.8	16.6	42.9 ±1.4	44.3 ±1.1	47.2 ±1.1	49.3 ±0.9
HypEmo	125M	97.6	585.8	1.2	**59.6** ±0.3	**61.0** ±0.3	**65.4** ±0.2	**66.3** ±0.2
Δ	-43.2%	-76.9%	-53.3%	-68.5%	+0.8%	+4.8%	+0.8%	+3.9%

Table 1: Left: efficiency comparison (lower is better), where ξ_{tr} denotes average training time for one epoch, ξ_{conv} denotes the average time to achieve the best results, and ξ_{inf} denotes the average inference time for the whole testing set. We evaluate efficiency for all systems on GE with batch size equals 16. Right: performance comparison (higher is better.) We conducted the experiment five times with different seeds, and report the average score with the standard deviation. The best (second-best) results are set in **bold** (underlined). "†" denotes reproduced results from official implementation, and Δ represents the efficiency and performance gain compared with LCL, the previous state-of-the-art, in relative percentage.

Dataset	Model	F1	F1 (+HypEmo)
ED	$\text{BERT}_{\text{base}}$	51.8	**57.7**
	$\text{RoBERTa}_{\text{base}}$	56.0	**61.0**
	$\text{ELECTRA}_{\text{base}}$	57.6	**58.9**
GE	$\text{BERT}_{\text{base}}$	62.9	**65.3**
	$\text{RoBERTa}_{\text{base}}$	64.0	**66.3**
	$\text{ELECTRA}_{\text{base}}$	64.1	**65.7**

Table 2: Performance in terms of weighted F1 score when HypEmo is added on different encoders.

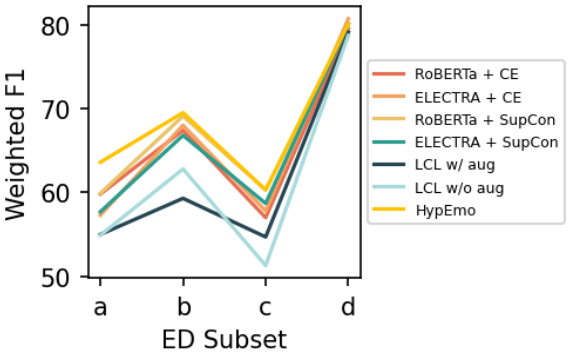
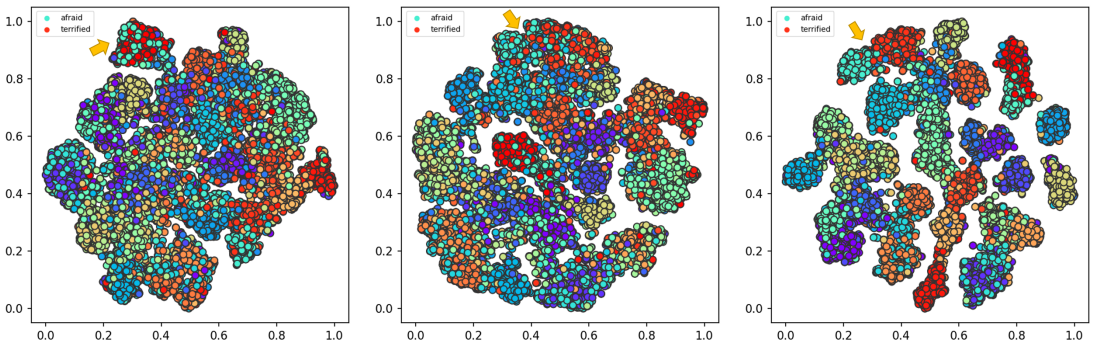


Figure 3: Case study for various ED subsets. We report F1 for brevity. CE is the standard cross entropy loss, and SupCon represents supervised contrastive loss (Khosla et al., 2020).

ELECTRA with standard cross entropy loss and supervised contrastive loss (Khosla et al., 2020). For supervised contrastive loss, we adopt back translation (Sennrich et al., 2016) to form the positive pairs, and we view it as a strong competitor because we expect contrastive loss helps to learn better representations could hence improve the FEC task. The result is shown in Fig 3. First, HypEmo outperforms all baselines by a large margin on the most difficult subset (a), which demonstrates the advantage of incorporating hyperbolic space when addressing fine-grained classification task. In particular, HypEmo beats the previous state-of-the-art, LCL, on this subset with a substantial improvement (54.9 v.s. 63.6.) Also, HypEmo surpasses models with cross entropy loss and supervised contrastive loss on a / b / c and comparably on d. As a / b / c are the top three most difficult sets, this result shows that label-aware hyperbolic weighted loss is conducive to separation under label sets which are more confusing. In addition, we compare with LCL with and without augmentation to ensure a fair comparison. The result shows that HypEmo consistently outperforms LCL even with data augmentation that doubles the size of samples. In summary, the proposed model performs the best when the label set contains confusing classes. For the most challenging sets, HypEmo outperforms models trained with conventional cross entropy loss and supervised contrastive loss, and even the state-of-the-art, LCL, while being more sample-efficient. When the label set is simpler, HypEmo performs on par with the others.



(a) Standard cross entropy loss (b) Supervised contrastive loss (c) Hyperbolic weighted loss

Figure 4: Representation generated from text encoder using different objective functions. We take *afraid* and *terrified* as examples for semantically close labels, and we highlight their position with the yellow arrow. Evidently, our proposed label-aware hyperbolic weighted loss leads to larger inter-class distances.

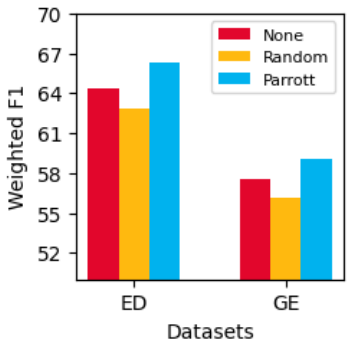


Figure 5: Label hierarchy method vs. performance.

4.5 Effect of Label Hierarchy

In this section, we investigate how the method used to form the label hierarchy affects performance. We compare three settings: *None*, *Random* and *Parrott*, as shown in Fig. 5. *None* means that we abandon all hierarchy and distribute labels uniformly on the Poincaré ball, and *Random* means that we randomly shuffle the correct order to make hierarchy meaningless. *Parrott* is the emotion model proposed by Parrott (2001), which defines a three-level emotion hierarchy. We observe that following an expert-defined hierarchy yields the best performance, better than without hierarchical information, and the worst is shuffling to discard all hierarchical information. Accordingly, we follow Parrott's emotion model to form the label hierarchy and learn the hyperbolic label embeddings.

4.6 Visualization of Representations

To better understand the effect of the proposed label-aware hyperbolic weighted loss, we compare representations learned with different commonly used objective functions. We trained on the ED dataset and projected the embeddings generated from the text encoder onto 2-dimensional space with t-SNE (Van der Maaten and Hinton, 2008). We compare standard cross entropy loss (Fig. 4a), supervised contrastive loss (Khosla et al., 2020) (Fig. 4b), and the proposed label-aware hyperbolic weighted loss (Fig. 4c). For supervised contrastive loss, we also used back translation as the augmentation to form positive pairs. Then, standard cross entropy loss was summed with supervised contrastive loss. As data augmentation doubles the data size, we expect the supervision and the objective to lead to more a separated representation. In Fig. 4a, we observe indistinct clustering, as many data points with different classes are mingled together, showing the difficulties of fine-grained classification in which different classes often share close semantics. In particular, *afraid* and *terrified* are confusing for models and their representations are mixed. With the use of supervised contrastive loss shown in Fig. 4b, the clustering becomes somewhat clearer but at the cost of increasing data. Last, in Fig. 4c, the inter-class distance is clearly larger than others, and the clusters are also more dispersed. Specifically, the representations of *afraid* and *terrified* are much more separated. This shows the advantage of the proposed label-aware hyperbolic loss, which yields better representations, even without the need for additional significant costs.

5 Conclusion

We propose HypEmo, a novel framework that includes a label-aware hyperbolic weighted loss to improve FEC task performance. By jointly learning the representations in Euclidean and hyperbolic

space, we leverage hybrid settings that combine the power of large-scale pre-trained language models and the mathematical characteristics of hyperbolic space to capture the hierarchical property of classes and the nuanced differences between them. With this design, the proposed method achieves state-of-the-art results in terms of weighted F1 on the GE and ED benchmark datasets. We show that the proposed model works even better when the labels are difficult to differentiate. Moreover, HypEmo outperforms methods that utilize data augmentation while being more efficient.

Limitations

Although the proposed framework yields promising results on two fine-grained emotion datasets—GoEmotions and Empathetic Dialogues—there remain limitations, including: (1) To the best of our knowledge, there is no such fine-grained emotion dataset in other languages. Although theoretically, our method should work fine on languages other than English, we can only show the results in English. (2) The proposed method works best when the label structure contains hierarchy, especially when the semantics of some labels are close and difficult to distinguish. When the label structure is flat and independent, our method may backoff to a conventional classification model.

Acknowledgement

This work is supported by the National Science and Technology Council (NSTC) of Taiwan under grants 111-2221-E-001-021 and 111-2634-F-002-022.

References

Gary Becigneul and Octavian-Eugen Ganea. 2019. Riemannian adaptive optimization methods. In *International Conference on Learning Representations*.

James W. Cannon, William J. Floyd, Richard Kenyon, and Walter R. Parry. 1997.

Ines Chami, Zhitao Ying, Christopher Ré, and Jure Leskovec. 2019. Hyperbolic graph convolutional neural networks. *Advances in neural information processing systems*, 32.

Soumya Chatterjee, Ayush Maheshwari, Ganesh Ramakrishnan, and Saketha Nath Jagarlapudi. 2021. Joint learning of hyperbolic label embeddings for hierarchical multi-label classification. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2829–2841, Online. Association for Computational Linguistics.

Boli Chen, Xin Huang, Lin Xiao, Zixin Cai, and Liping Jing. 2020. Hyperbolic interaction model for hierarchical multi-label classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7496–7503.

Hyunghoon Cho, Benjamin DeMeo, Jian Peng, and Bonnie Berger. 2019. Large-margin classification in hyperbolic space. In *The 22nd international conference on artificial intelligence and statistics*, pages 1832–1840. PMLR.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *ICLR*.

Dorottya Demszky, Dana Movshovitz-Attias, Jeongwoo Ko, Alan Cowen, Gaurav Nemade, and Sujith Ravi. 2020. GoEmotions: A dataset of fine-grained emotions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4040–4054, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Bhuwan Dhingra, Christopher Shallue, Mohammad Norouzi, Andrew Dai, and George Dahl. 2018. Embedding text in hyperbolic spaces. In *Proceedings of the Twelfth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-12)*, pages 59–69, New Orleans, Louisiana, USA. Association for Computational Linguistics.

Paul Ekman. 1999. *Basic Emotions*. Handbook of Cognition and Emotion.

Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. 2018. Hyperbolic neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 5350–5360.

Dingkun Long Guangwei Xu Ning Ding Haoyu Zhang Pengjun Xie Gongshen Liu Jie Zhou, Chunping Ma. 2020. Hierarchy-aware global model for hierarchical text classification.

Hamed Khanpour and Cornelia Caragea. 2018. Fine-grained emotion detection in health-related online posts. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1160–1166, Brussels, Belgium. Association for Computational Linguistics.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 18661–18673. Curran Associates, Inc.

Qi Liu, Maximilian Nickel, and Douwe Kiela. 2019a. Hyperbolic graph neural networks. *Advances in Neural Information Processing Systems*, 32.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach.

Federico López and Michael Strube. 2020. A fully hyperbolic neural model for hierarchical multi-class classification. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 460–475, Online. Association for Computational Linguistics.

Dheeraj Mekala, Varun Gangal, and Jingbo Shang. 2021. Coarse2fine: Fine-grained text classification on coarsely-grained annotated data.

Maximillian Nickel and Douwe Kiela. 2017. Poincaré embeddings for learning hierarchical representations. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

W Gerrod Parrott. 2001. *Emotions in social psychology: Essential readings*. psychology press.

W. Peng, T. Varanka, A. Mostafa, H. Shi, and G. Zhao. 2021. Hyperbolic deep neural networks: A survey. *IEEE Transactions on Pattern Analysis Machine Intelligence*, (01):1–1.

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.

Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Eric Michael Smith, Y-Lan Boureau, and Jason Weston. 2021. Recipes for building an open-domain chatbot. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 300–325, Online. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Varsha Suresh and Desmond Ong. 2021. Not all negatives are equal: Label-aware contrastive loss for fine-grained text classification. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4381–4394, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Da Yin, Tao Meng, and Kai-Wei Chang. 2020. SentiBERT: A transferable transformer-based architecture for compositional sentiment semantics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3695–3706, Online. Association for Computational Linguistics.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
After conclusion (section 6).

☑ A2. Did you discuss any potential risks of your work?
Included in the limitations section.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Left blank.

☑ A4. Have you used AI writing assistants when working on this paper?
Grammarly, correct grammar errors.

B ☒ Did you use or create scientific artifacts?

Left blank.

☐ B1. Did you cite the creators of artifacts you used?
Not applicable. Left blank.

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Not applicable. Left blank.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Not applicable. Left blank.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. Left blank.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Left blank.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Left blank.

C ☑ Did you run computational experiments?

Section 5.

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Section 5.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Section 5.

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Section 5.

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Section 5 and the provided code.

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Not applicable. Left blank.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Not applicable. Left blank.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Not applicable. Left blank.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. Left blank.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.