Graduation Thesis Final Report

Multimodal Semi-supervised Learning for Sentiment Analysis of Image Macros

Students	1. **Pham Thai Hoang Tung (HE141564)** Bachelor of Computer Science 2. **Ngo Tien Anh (HE141442)** Bachelor of Computer Science 3. **Nguyen Tan Viet (HE153763)** Bachelor of Artificial Intelligence
Supervisor	**Associate Professor Phan Duy Hung**

Hoa Lac Campus – FPT University
December 11, 2022

Acknowledgment

This thesis is the result of our hard-working after a long time. We thank Associate Professor Phan Duy Hung for always being by our side and supporting us in completing this thesis. We also express our sincere thanks to FPT University for providing valuable resources and timely assessments to make the thesis as complete as it is today. We also would like to express our deep gratitude towards our supporters, Mrs. Nguyen Thi Chuong, ILotusLand VietNam, Ho Chi Minh City, Vietnam, and Dr. Le Bin Ho, Tohoku University, Sendai 980-8579, Japan, for helping us with data storage and hardware resource when we needed the most. In addition, each of us would like to thank the rest of our teammates, for all their efforts. For without all of those efforts, assistance, and guidance, this thesis wouldn't have been completed.

Although we tried very hard, we still needed to avoid mistakes. The team is happy to hear additional comments and reviews.

Thank you very much!

"Dreams do come true, if only we wish hard enough. You can have anything in life if you will sacrifice everything else for it."

— J.M. Barrie, Peter Pan

Abstract

Memes in the form of image macro are a part of social media content nowadays. The meme usually has an underlying meaning that needs to be sentiment analyzed for censoring harmful content. Meme censoring systems by machine learning raise the need for a semi-supervised learning solution to leverage a massive quantity of unlabeled memes on the internet and reduce the difficulties of the annotation process. Moreover, the machine learning approach should utilize multimodal data because a meme's meaning usually comes from both visual and linguistic. Therefore, in this research, we proposed a multimodal semi-supervised learning approach that outperformed other multimodal semi-supervised learning and supervised learning SOTA when comparing the result on the Multimedia Automatic Misogyny Identification (MAMI) dataset of the meme. Besides successfully applying other excellent studies about multimodal data and imbalanced data, such as CLIP and distribution balanced loss, our research presents a new training manner that wisely combines auto-encoder and classification tasks to utilize unlabeled data.

Warning: Images in this document might contain offensive language.

Table content

Annotations

List of Figures

List of Tables

Chapter 1 Introduction

1.1 Problems

1.1.1 Basic Concepts

In recent years, along with the development of social networks, Memes have gradually become one of the most popular tools for expressing emotions and communicating. So, what are memes?

Historically, the concept of "meme" first appeared in 1976 in the book The Selfish Gene (Figure 1) by Richard Dawkins (a British author). He used the word "meme" to talk about ideas and behaviors that were spread in the community.

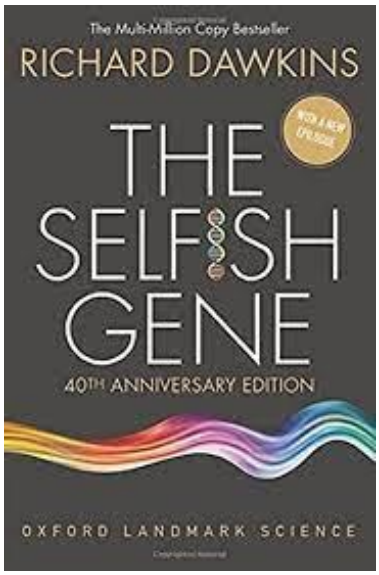

Figure 1. The book The Selfish Gene

However, today the concept of "meme" has been used everywhere on social networks. Thus, was born a new concept called "Internet meme."

An "Internet meme," often abbreviated as "meme," is an idea, a famous saying, a trend, or a behavior that is spread on the internet [1]. Nowadays, the term is applied to many structures, such as challenges, GIFs, videos, and viral sensations. Concept memes can vary between online communities and change over time. The use of memes for such purposes has been widely recognized and has gone viral on various media platforms.

Figure 2. Example Meme (from left to right: only image; image and text)

Since the meme has quite a lot of structure, the study of its semantics becomes scattered. Therefore, our team decided to focus their research on **"Image macro"** - a term derived from the Something Awful forum [28]. "Image macro" is the most common form of internet meme; it consists of an image and a short piece of text overlaid on top of the background. To understand the meaning of Image macro, it is sometimes necessary to understand both the meaning of the image and the text and then combine them.

In summary, to simplify the concepts in the research, from here on, we would like to refer to "Image macro" as "meme" for short.

1.1.2 Issues surrounding hateful content on social media

Due to the widespread popularity of memes, many are created that are no longer intended to be amusing or to show humor but also to create irony or harmful content such as discrimination, race, gender, religion, or even political issues. Therefore, social media platforms such as Twitter, Facebook, and Instagram are very interested in the matter of "emotion analysis in memes" so that they can prevent memes with harmful content.

In the past, harmful content was often presented only through text, and its detection was relatively easy through textual research, computational methods, and Natural Language Processing (NLP). However, when harmful content is hidden in memes, it will be quite challenging to detect. Let us look at the following two examples in Figure 3.

Figure 3. Example Meme (image + text)

In the examples in Figure 3 above, we can see that detecting harmful content through the "meme sentiment analysis" method is quite complicated because it has to understand and combine the visual and textual content of the meme. Even humans take time to think to understand, so applying those methods to machine learning is even more difficult. In addition, data manually labeled by humans will not be comprehensive and subjective (of the labeling participants), and sometimes there will be conflicts and arguments because each person will have an opinion and understanding that differ about the content of a meme [9, 11, 31, 34, 38]. A semi-supervised learning approach on multimodal data containing images and text can cut down the struggle of the annotation process.

1.2 Related works

In 2022, Kumar and Nanadakumar proposed the HateCLIpper architecture, which models using a Contrast Language-Image Pre-Training (CLIP) encoder [32] through a matrix feature interaction (FIM) [22]. Based on the FIM representation, this model achieved the highest performance on the Hateful Memes Challenge (HMC) dataset [20] with an AUROC index of 85.80. In that warehouse, the human performance index is only 82.62. Back in 2020, most of the top solutions of HMC were based on VisualBERT [23], a popular backbone for Vision and Language tasks [19]. In other competitions about meme sentiment analysis [9, 34], winners usually build their approach are supervised learning based on VisualBERT or CLIP [9, 30].

There are some studies about semi-supervised learning around multimodal content, including images, text, and other modalities. Their tasks vary. Hu et al. project their modality to a feature on embedding space and perform cross-modal retrieval tasks, i.e., retrieving text by image and vice versa [17]. Another research by Sunkara et al. employs a large unlabeled

text corpus and extensive unlabeled audio data to pre-train modality encoders, then fusion output of the encoders to train punctuation prediction in conversational speech [40]. For the classification task, Liang et al. do emotion recognition on video by extracting visual, acoustic, and lexical signals from both labeled and unlabeled videos [24]. Their model is trained end-to-end, combining two tasks concurrently. One task is auto-encoder on entire data, in which each modality tries to reconstruct itself, and the other task is emotion classification on latent representations of labeled data's modalities. Although these studies perform different tasks or modalities from our work, they inspired us on ways to do semi-supervised learning on multimodal data.

There was a SOTA done by Yang et al. about implementing semi-supervised classification on images and text in 2019 [45]. They proposed Comprehensive Semi-Supervised Multimodal Learning (CMML), which utilizes unlabeled data to strike a balance between consistency and divergence among modalities by introducing diversity and consistency measurements. Optimizing diversity measurements can increase diversity among modalities' predictions, while consistency measurements minimize the disagreement among them. CMML achieves competitive results on various large-scale multimodal datasets such as FLICKR25K [18], IAPR TC-12 [8], NUS-WIDE [2], and MS-COCO [25]. However, this method is hard to optimize where the loss function constitutes multiple supervised losses and regularized unsupervised losses.

In domain meme, some researchers tried to use data on tasks that do not require labels. Sharma et al. create a pre-train model by self-supervised learning on a collection of public meme datasets [39]. Gunti et al. tried to embed images and words in the same space by training a Siamese network that receives a pair of image-word belonging to a meme [13]. As a result, they make image embedding of a meme have a semantic meaning driven by word embedding. These studies show how valuable unlabeled meme data can be used.

1.3 Motivation

Based on the problems of using machine learning on hateful multimodal social media content and earlier research, two issues can be identified:

- Firstly, the data collected on the meme to serve the research of combining images and text is still small; on the other hand, the cost of hiring labor to re-label is relatively high.
- Second, there are now studies on this issue, mainly results from competitions. However, most research groups go toward supervised learning.

Therefore, our team realizes that the current urgent problem is to solve the problem "Analysis of emotions in memes to detect harmful content" in the direction of research

"Multimodal semi-supervised learning." Specifically, in this study, we will use the MAMI (Multimedia Automatic Misogyny Identification) dataset [9] to identify and identify memes with inappropriate and sexist content against women (details of the dataset will be explained in more detail in Chapter 3).

1.4 Contribution

In this thesis, we created a multimodal semi-supervised learning approach. Our contributions can be outlined as follows:

- We propose a pre-train model based on the extracted features from CLIP to specifically pre-train on small datasets, namely Cross Modality Auto Encoder (CROM-AE), without requiring labels.
- A custom task-specific supervised model is created to incorporate the CROM-AE encoder's features and CLIP's features (RAW-N-COOK).
- We apply an efficient supervised loss to solve the multi-label problem and class imbalance of the meme dataset.

Chapter 2 Background

2.1 Transformer architecture

The transformer is a powerful architecture and was first introduced in [42]. Since its first appearance, researchers have developed a variety of variants that have achieved SOTA in various fields, such as natural language processing, computer vision, and speech recognition. The basic block of the transformer is the attention formulas, defined as:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (1)$$

Where Q, K, and V are Query, Key, and Value matrices, respectively, d_k is the dimension of Key and Query. The attention kernel tries to capture the interaction between Query and Key matrices normalized by its square root dimension which is converted into probability format by the SoftMax function. The SoftMax term works as a series of attention weight vectors that acts upon the Value matrix. In a basic transformer block, there would be multiple attention functions called multi-head attention to capture different views of input.

$$MultiHead(Q, K, V) = Concat(head_1, \dots, head_n)W^O \quad (2)$$
$$\text{where } head_i = Attention\left(QW_i^Q, KW_i^K, VW_i^V\right)$$

The transformer block further consists of a layer norm, a position-wise feed-forward network that comprises 2 linear layers and an activation ReLU in between, and residual connections for ease of gradient flowing. The architecture we use is a modified transformer in [33] with the layer norm being the first layer. This block can be visualized in Figure 4.

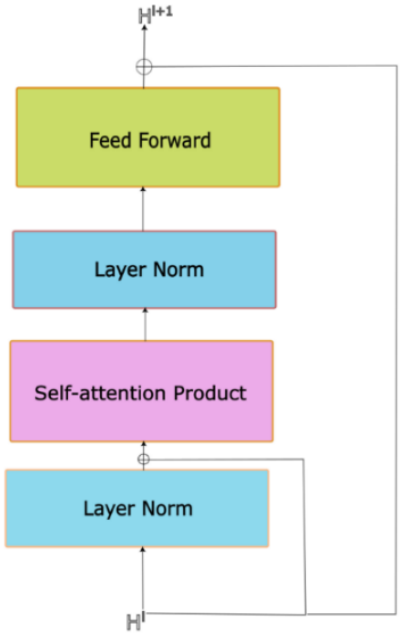

Figure 4. The basic transformers block with modifications by [33]

2.2 Language models

Natural language processing has seen a giant leap in SOTA's achievement thanks to the transformer framework. [4] introduced a pre-trained encoder comprising a 12-layer transformer called bi-directional encoder representations for transformers (Bert). Since then, there have been many Bert-based pre-trained models with different architectures and pre-training approach modifications, such as Roberta [26], Distill-Bert [37], Electra [3], and Deberta [16]. Nevertheless, the transformations of text to language model remain essentially the same. First, the sub-words of text data will be converted to their unique token ids, the model will perform look-ups of their corresponding token and positional embedding which will be added together and sent to the transformer encoder. Following [4], the output [CLS] representation of the transformer encoder is chosen to fine-tune task-specific datasets. This process can be visualized in Figure 5.

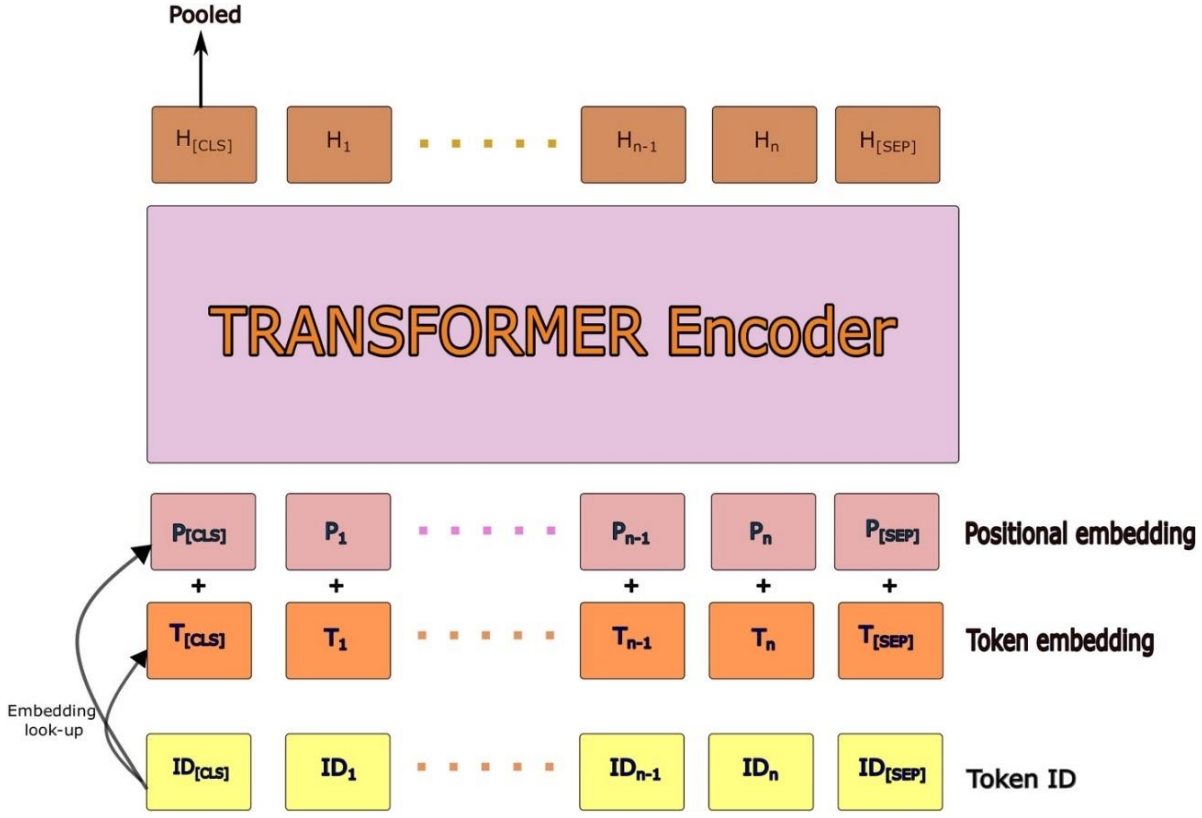

Figure 5. Overall architecture of language models

2.3 Vision Transformer

As described in 2.1 above, Transformer has dominated the NLP field recently, but not for computer vision tasks. In 2021, Google Brain successfully invented a new architecture from a pure Transformer named Vision Transformer (ViT) without relying on CNN [5]. When pre-

trained on vast amounts of data (approximately 303M images), ViT shows its excellence in transferring to medium and small datasets of image classification, compared to SOTA CNN trained with the same amount of data.

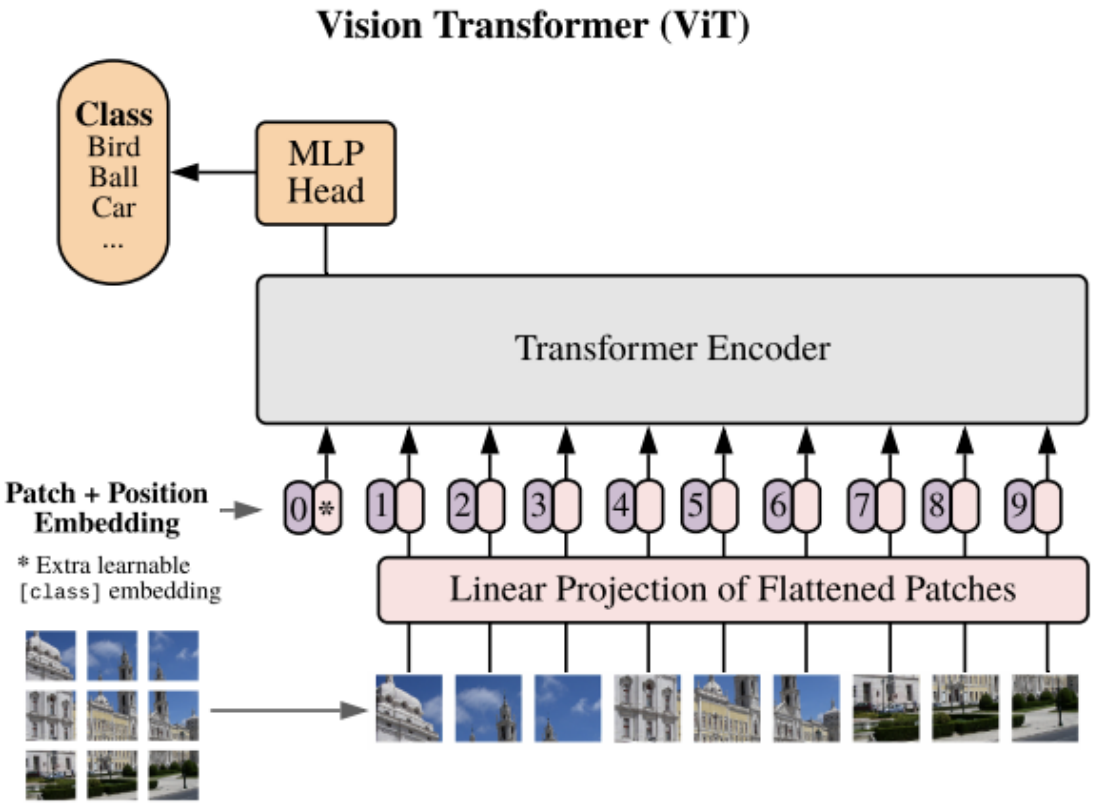

Figure 6. ViT Architecture [5]

As described in Figure 6, in ViT, each image is treated as a sequence of words by being divided into fixed-size patches. Then, patches are linearly projected, plus positional embedding to create new corresponding vectors. An extra-learnable [class] token is also initialized and added to the sequence. These vectors are fed into the standard Transformer Encoder, which includes a stack of Multi-head Attention layers, and outputs the vector of [class] token to learn the classification task.

2.4 Multimodal

Contrastive Language-Image Pre-training (CLIP) [32] is a large-scale visual linguistic pre-trained model on the WebImageText dataset (WIT). The task-agnostic objective of CLIP is based on a simplified version of ConVIRT [46] which maximizes the real pair on the diagonal entries and minimizes non-diagonal entries of the similarity matrix.

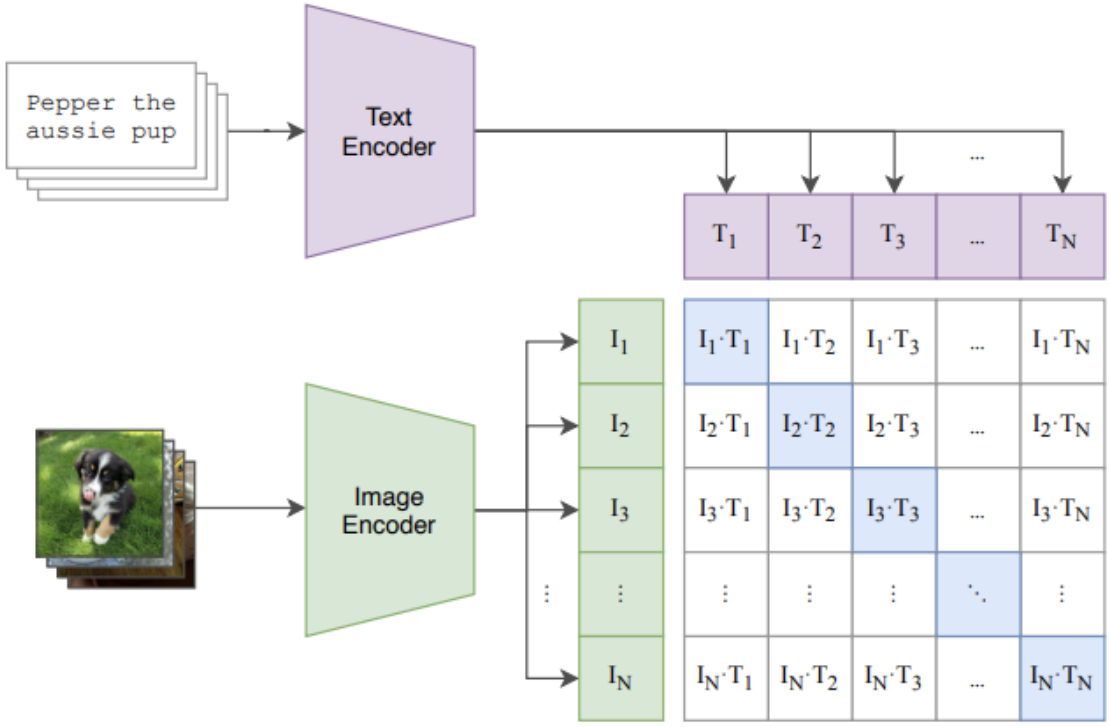

Figure 7. CLIP overview, the picture was taken from [32].

Using this pre-training approach, CLIP can learn to associate visual concepts in natural language to images more flexibly, unlike conventional computer vision pre-trained models which have to rely on annotated large-scale datasets including just a few hundred visual concepts such as ImageNet [36]. Moreover, CLIP is pre-trained on the web-crawled dataset. Thus, it may have learned popular concepts of visual and linguistic features on the internet, including memes. For these reasons, CLIP is a very bright candidate for our study on multimodal image macros in the wild.

Because OpenAI-CLIP has many pre-train encoder versions, one can extract their features via the version of the transformer as follows:

For Image: Features are extracted from Vision-Transformer Encoder [5]. As described in 2.3, an image will be divided into n patches p x p, where p is the edge size of one patch. The resulting patches will be flattened to create n flattened patches P with $P \in \mathbb{R}^{P \times P}$.

These P patches will go through the embedding layer and transformer encoder of ViT as:

$$P \oplus n \xrightarrow{embedding} E_{image} \oplus n \xrightarrow{encoder} H_{image} \oplus n \quad (3)$$

Where $E \in \mathbb{R}^d$ is the embedding output, $H \in \mathbb{R}^d$ is the output of the transformer encoder, $\oplus$ is the concat operator, d is the embedding dimension, which is usually set to 768.

To produce the output of the image encoder. CLIP takes the first patch output of the encoder.

$$F_{image} = H_{image}{}_{[0]}^{\oplus n} \quad (4)$$

For text: The procedure for the text transformer is also similar. Given a sequence $S = \{w_1, w_2, \dots, w_m\}$ with $S \in \mathbb{R}^{m \times |V|}$, V is the vocab of model, m is the length of sequence. S will go through embedding layer and encoder layer as demonstrated with CLIP image encoder above in (3), to achieve $H_{text}{}^{\oplus m}$. For text data, CLIP will take embedding of the last word ([EOS] token) to be the output of CLIP text.

$$F_{text} = H_{text}{}_{[EOS]}^{\oplus m} \quad (5)$$

2.5 Semi-supervised learning

Semi-supervised learning (SSL) is a type of learning that falls between supervised and unsupervised learning. In addition to unlabeled data, the algorithm is given some supervision information linked with some samples. In other words, the data set $X = (x_i); i \in [n]$ may be separated into two subsets: the small set of points $X_l = (x_1, \dots, x_l)$ for which labels $Y_l = (y_1, \dots, y_l)$ are supplied, and the larger scale set of points $X_u = (x_{l+1}, \dots, x_{l+u})$ for which labels are unknown. SSL aims to minimize the following loss function [44]:

$$L = L_s + \alpha L_u \quad (6)$$

where L_s is the supervised loss and L_u is the unsupervised loss.

With a dataset D includes a labeled part D_l and an unlabeled part D_u, the objective of SSL algorithms is to utilize unlabeled samples in D_u so that SSL obtains a more accurate model than the supervised model only trained on D_l. For example, we could get information about overall data distribution from D_u to estimate better the decision boundary between classes as in Figure 8 below.

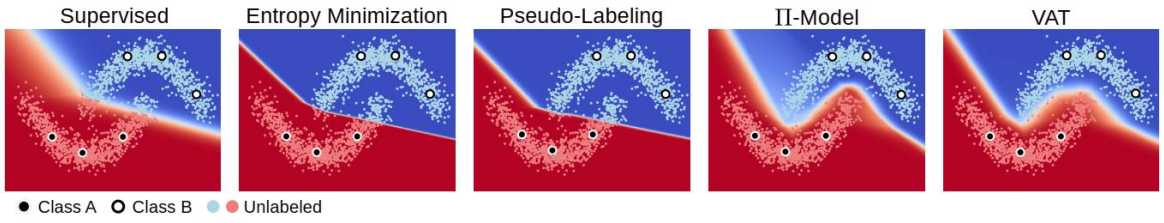

Figure 8. The decision boundaries of supervised and different SSL algorithms on a two-moons shape dataset, with 6 labeled samples, 3 for each class, and the remaining points as unlabeled data [29].

Another example of how SSL can help build better models is document classification, where each text document is assigned to a specific topic [6]. Assuming we have a small labeled set, our document is presented by the combination of words, and the collection of documents in a topic usually contains characteristic words that other topics rarely have. For example, the

topic "physic" can appear in the words: "neutron", and "quark" while the topic "biology" can contain the words: "evolutionary", and "organism". We build a machine learning model to classify documents based on characteristic words. Due to the small labeled training set, some characteristic words which should have appeared in the data may not appear. For instance, no document in the training set has the words: "chromosome", and "mutation". If a test "biology" document only contains characteristic words absent in the training set, the machine learning model fails to categorize this document. It is when SSL with unlabeled data can help. Consider a large-scale unlabeled dataset in which characteristic words of a topic usually occur concurrently. For example, words like "chromosome" and "mutation" appear with "gene", and "gene" usually appears with "evolutionary". Then we can add this information to guide the document classification model to achieve a better result on the document containing characteristic words missing in the labeled training set.

Formally, suppose the underlying marginal data distribution p(x) over the input space contains information about the posterior distribution p(y|x). In that case, one might leverage unlabeled data to gain information about p(x), thereby p(y|x) [47]. L_u in (6) is used to optimize the process of finding information in p(x) relevant to p(y|x).

Although almost machine learning problems meet the condition that p(x) holds information of p(y|x), how p(x) and p(y|x) interact are usually different. Therefore, SSL approaches vary. In the taxonomy of SSL, one group of SSL methods can be defined as unsupervised preprocessing [6]. In unsupervised preprocessing methods, labeled and unlabeled data are used in two separate stages. The first one typically leverages an unsupervised task, such as clustering or auto-encoder, to preprocess data or pre-train a model, then the second stage use data or model manipulated in the previous stage to learn directly on the main task, such as classification [10] [7].

2.6 Autoencoder

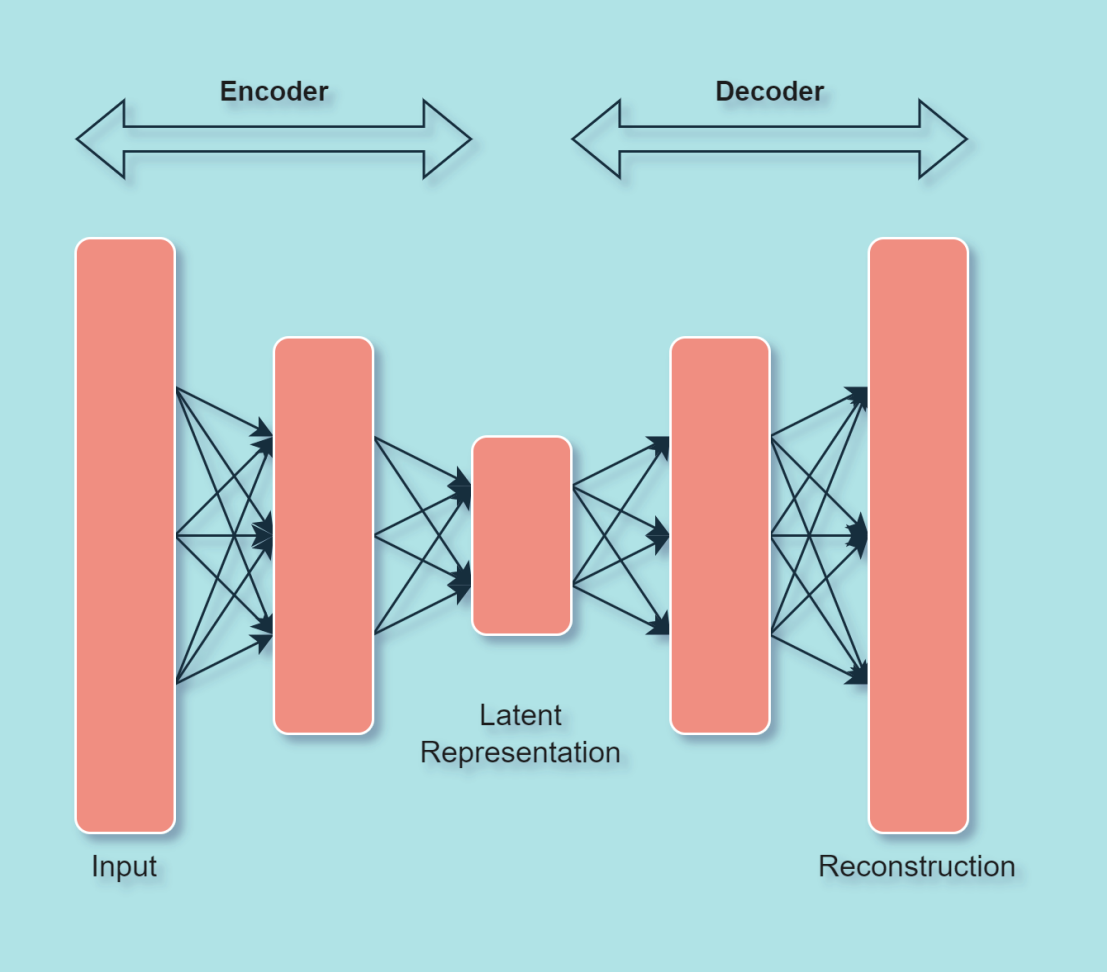

Figure 9. Autoencoder Architecture

An autoencoder is a neural network that has been taught to replicate its input to its output. The network may be divided into two parts: an encoder function h = f(x) maps input to its latent representation h, and a decoder function r = g(h) maps h to a reconstruction r. The general architecture of an autoencoder can be seen in Figure 9. If an autoencoder copies every input perfectly, i.e., r = x for all x, it seems useless [12]. Autoencoder is designed to approximate input or to prioritize which aspects of the input should be reconstructed, hence filtering meaningful properties from training data. Therefore, after training an autoencoder model, the model can extract helpful features from the input by taking the result on h.

Autoencoder can train with the same techniques as a default neural network, with a defined loss function (usually mean-square-error) minimized by an optimizer such as minibatch gradient descent via backpropagation. The loss function, also called the reconstruction error, measures the difference between reconstructed data and input.

As the training process does not involve any label, an autoencoder can be used to utilize unlabeled data for semi-supervised learning.

2.7 Loss function for multi-label imbalanced classification

For multi-label datasets, one possible choice for supervised loss function is binary cross entropy loss (**BCE**) for N samples which is defined as:

$$BCE = -\frac{1}{N}\sum_{i=0}^{N} y_i \cdot \log(p) + (1 - y_i) \cdot \log(1 - p) \quad (7)$$

Where p is the prediction probability of the model, y_i is a ground truth of a sample. BCE loss optimizes each label independently and does not consider the dependent co-occurrences of labels in each sample. Furthermore, this loss is symmetric; therefore, negative labels and positive labels will be treated the same which will lead to over-suppression on the negative sides [35], in other words, it will suffer from low confidence predictions, thus decreasing the recall of the models.

To alleviate it, [35] uses asymmetric loss that introduces a probability margin parameter to shift the negative distribution, they introduce two separate gamma coefficients: γ_-, γ_+; and a shifted probability by a constant for negative loss. The shifted probability can prevent the prediction of negative labels with low confidence to be dominant in the loss, while different gamma coefficients can treat the positive class and negative class differently due to the nature of imbalance data, thus they set $\gamma_- \geq \gamma_+$ to account for more contribution of positive labels. However; this loss does not handle the multi-label nature of data where the sampling rate of positive classes is not the same as the sampling rate of an instance having those classes.

[43] tackles these issues on logits prediction level by introducing a scale factor on negative logits, namely rebalanced distribution loss. **Re-balanced distribution loss** can be formalized as follows:

$$\mathcal{L}_{DB}(x^k, y^k) = \frac{1}{C}\sum_{i=0}^{C} \hat{r}_i^k \left[y_i^k \log\left(1 + e^{-(z_i^k - \nu_i)}\right) + \frac{1}{\lambda}\left(1 - y_i^k\right) \log\left(1 + e^{\lambda\left(z_i^k - \nu_i\right)}\right) \right] \quad (8)$$

Where $\hat{r}$ is the re-balanced weighting factor on the train labeled set and is calculated by taking the ratio of expectation of class-level sampling frequency versus instance-level sampling frequency and being scaled to proper range as shown in (12),

$$P_i^C(x^k) = \frac{1}{C}\frac{1}{n_i} \quad (9)$$

$$P^I(x^k) = \frac{1}{C}\sum_{y_i^k=1}\frac{1}{n_i} \quad (10)$$

$$r_i^k = \frac{P_i^C(x^k)}{P^I(x^k)} \quad (11)$$

$$\hat{r} = \alpha + \frac{1}{1+e^{-\beta\times(r-\mu)}} \quad (12)$$

, λ is the scale factor that affects the strength of zero-suppression with respect to negative logits and ν is the class-specific bias of the model. In the multi-label setting, the sampling rate of one positive class will depend on other class labels, distribution-balanced loss handle this by balancing the weight of each class and their conditional couplings with other classes on the instance level. In our implementation, we use a focal version of this loss, we set p_+ as predictions of positive logits and p_- as prediction of negative logits, we have:

$$p_+ = \frac{1}{1+e^{-(z_i^k-\upsilon_i)}} \quad (13)$$

$$p_- = \frac{1}{1+e^{-\lambda(z_i^k-\upsilon_i)}} \quad (14)$$

DB loss can be re-written:

$$\mathcal{L}_{\mathrm{DB}}(x^k, y^k) = -\frac{1}{C}\sum_{i=0}^{C}\hat{r}_i^k\left[y_i^k\log(p_+) + \frac{1}{\lambda}\left(1-y_i^k\right)\log(1-p_-)\right] \quad (15)$$

Focal-DB loss can be written as:

$$\mathcal{L}_{DB-focal}\left(x^k, y^k\right) = -\frac{1}{C}\sum_{i=0}^{C}\hat{r}_i^k\left[(1-p_+)^\gamma y_i^k\log(p_+) + \frac{1}{\lambda}(p_-)^\gamma\left(1-y_i^k\right)\log(1-p_-)\right] \quad (16)$$

We set γ= 2.

2.8 Activation function

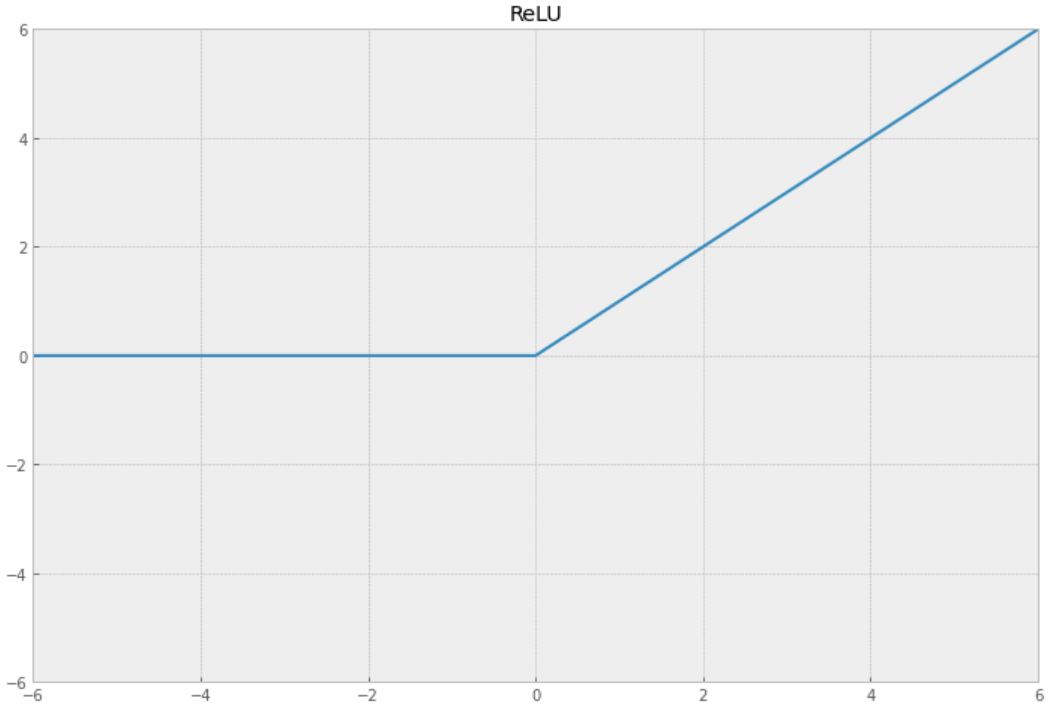

Figure 10. A plot of Activation Function ReLU

Rectified Linear Unit (ReLU) activation function has the formula:

$$ReLU(x) = \max(0, x)$$

Or

$$ReLU(x) = \begin{cases} x, & if\ x \geq 0 \\ 0, & otherwise \end{cases}$$

The ReLU function has been extensively used when training neural networks recently. ReLU function filters value smaller than 0. Its graph can be seen in Figure 10. ReLU has fast convergence speed and rapid computing. However, ReLU also has a disadvantage: nodes with a value less than 0 through ReLU activation will become 0. If the nodes are converted to 0, then it will not make sense for the linear activation step in the next layer. The problem is called "Dying ReLU" [27]. There is another activation function that can solve this case named PReLU.

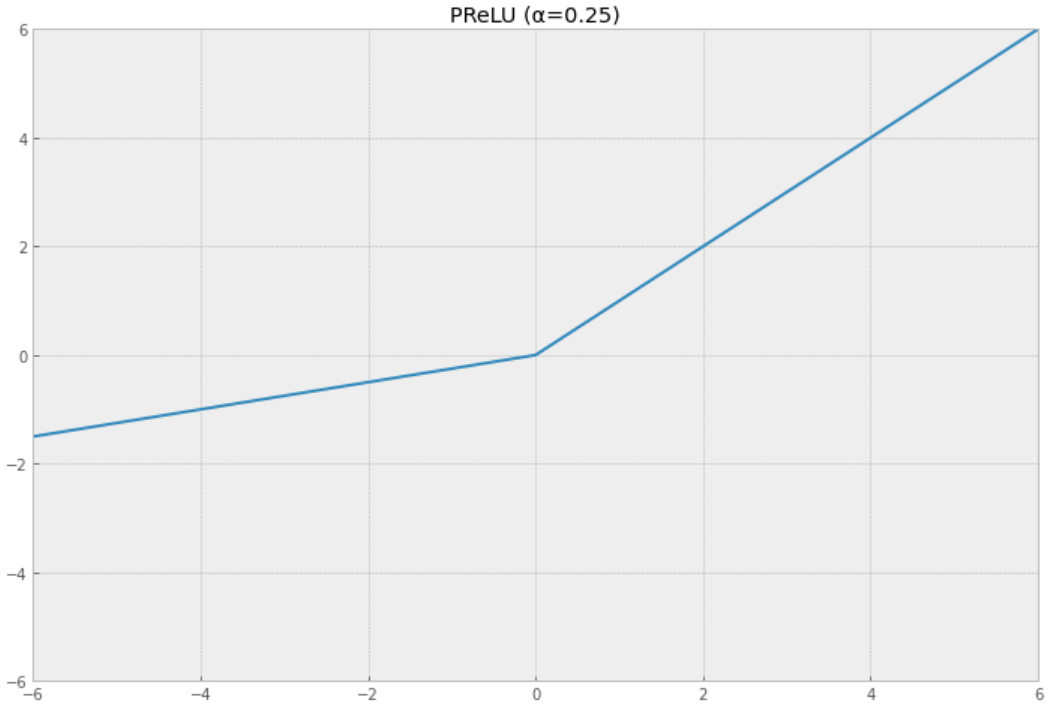

Figure 11. A plot of PReLU with α=0.25

Parametric Rectified Linear Unit (PReLU) has the formula:

$$PReLU(x) = \max(0, x) + \alpha \min(x)$$

Or

$$PReLU(x) = \begin{cases} x, & if\ x \geq 0 \\ \alpha x, & otherwise \end{cases}$$

Where α is a learnable parameter controlling the slope of the negative part. α is usually initialization equal to 0.25. Its graph can be seen in Figure 11.

PReLU was introduced in 2015 as the first time a machine-learning model surpassed human-level performance in the image classification task [15]. PReLU is a non-linear activation that does not evaporate all negative values of the previous layer. It scales negative values by an adaptive learnable rate α, while keeping all positive values unchanged, as ReLU does on the positive. As a result, PReLU helps the optimizer update weights linked with negative values and allows a model layer to output meaningful negative values.

Another minor modification of PReLU has a formula:

$$PReLU(x_i) = \begin{cases} x_i, & if\ x_i \geq 0 \\ \alpha_i x_i, & otherwise \end{cases}$$

Where $x_i \in x$ is an entry depth dimension of a layer. Each entry has its scale parameter α_i.

Chapter 3 Data

3.1 Overview

In today's developed society, women are equal to men. However, in reality, many places still exist where women are often oppressed and discriminated against according to ancient customs. Especially on social networking sites or sites that discriminate against women appear in memes with offensive content. Also, when misused, memes can amplify gender stereotypes and gender inequality. Therefore, a contest was created to detect inappropriate content and prevent this reality. The contest is named Multimedia Automated Misogyny Identification (MAMI) [9]. Use both available images and text to identify inappropriate memes for women. The contest task for their MAMI dataset consists of two main functions:

Task A: Identify memes with hateful content. The memes will be classified as either hate women or not hate women. The task is equivalent to a binary classification task.

Task B: Identify memes with misogynistic content by incorporating identification of categories such as stereotyping, shame, protest, and violence. The task is equivalent to a multi-label classification task with 4 binary labels.

In this study, we only do the research on the task B of MAMI dataset. This dataset consists of 10K memes for training and 1K memes for testing. We further split the training set into 3 sets including the labeled set, unlabeled set, and validation set. We reserve 2K samples for the validation set, the rest of 8K will be divided into labeled and unlabeled data based on a ratio p:1-p. We will vary p to verify the power of our method on multiple labeled data ratios. The detail of chosen value p is in 5.2.

An example of provided metadata denoting meme file and the corresponding text, label is illustrated in Figure 12. Some examples of memes in MAMI dataset are visualized in Figure 13. The meaning of each in columns in metadata is shown in Table 1. Some statistics of text in the dataset are presented in Figure 14, Figure 15, Figure 16.

	file_name	Text Transcription	misogynous	shaming	stereotype	objectification	violence
0	5532.jpg	My Mom telling me behave or I'll end up like t...	0	0	0	0	0
1	3144.jpg	1511 BestDemotivationalPosters.com CORN DOG? Y...	1	0	1	1	0
2	7755.jpg	A PROSTITUTE GAVE ME ALL MONEY SHE MADE FOR TH...	0	0	0	0	0
3	4959.jpg	IM GONNA SLAP ÎŸÎ¤Î— SAN ANICE PACK ON RIHANNA...	1	0	0	0	1

Figure 12. MAMI metadata

Figure 13. Some examples of memes that used in the MAMI dataset

Table 1. Meaning columns in metadata of MAMI dataset

Column Name	Meaning
file_name	Name of image file
Text Transcription	Extracted OCR Text in Meme
misogynous	Memes are classified as misogynous ("1") or not misogynous ("0"). Used for task A.
shaming	Memes are classified as shaming ("1") or not shaming ("0"). Used for task B.
stereotype	Memes are classified as a stereotype ("1") or not stereotype ("0"). Used for task B.
objectification	Memes are classified as objectification ("1") or not objectification ("0"). Used for task B.
violence	Memes are classified as violent ("1") or not violent ("0"). Used for task B.

Figure 14. Word cloud in training dataset

Figure 15. Histogram length of Text

Figure 16. Example meme with short text in MAMI

One problem we meet in MAMI dataset is class-imbalance, which is also usually met in other meme datasets [20, 31, 34, 38]. The sample of each class in a label is shown in Figure 17.

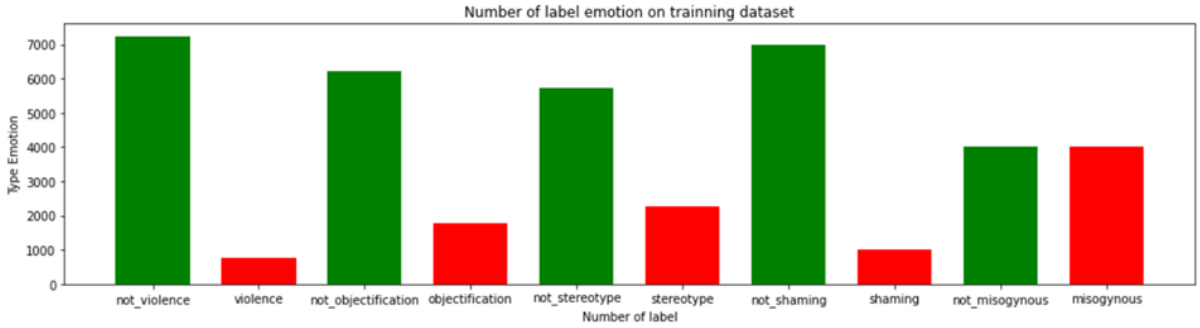

Figure 17. Number of samples by category

3.2 Preprocessing

As the text comes from a real-world dataset, we have processed them as follows: remove URL mixed-in text, remove non-ASCII characters, convert all characters to lowercase, remove punctuation. For the image, we resized it to a square image with the size of 224x224 to fit the size of the pretrain model.

Chapter 4 Methodology

4.1 Overview

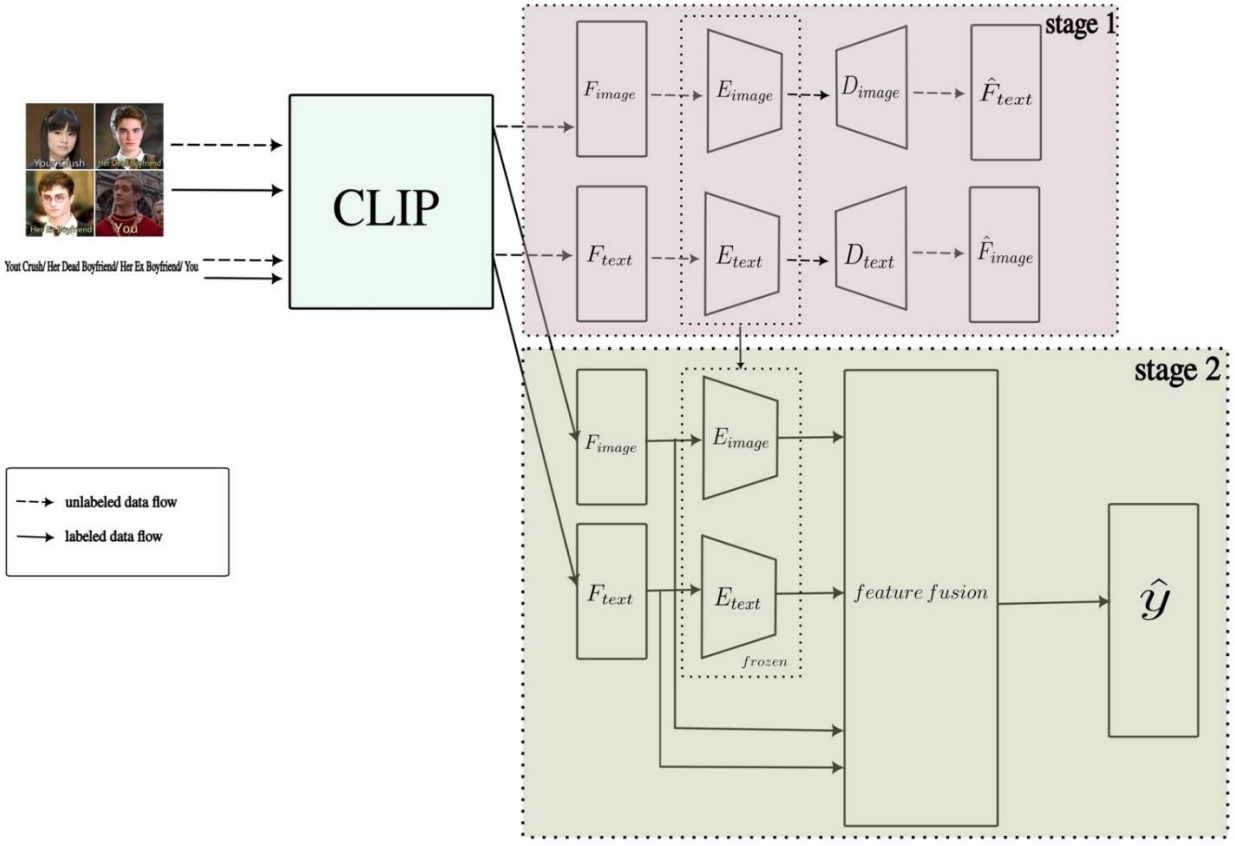

Figure 18. The overall of our SSL approach

As described in Figure 18, our SSL approach can be presented as follows. Firstly, a pair of (image, text) will be fed into the image encoder and text encoder of the CLIP model to extract a pair of CLIP feature vectors (F_{image}, F_{text}). Then, we do two sequentially stages as follows:

Stage 1 - Unsupervised Pre-training: We train Cross Modality Auto Encoder (CROM-AE), which takes the CLIP feature of one modality to predict the CLIP feature of the remaining modality, i.e., the image feature F_{image} tries to predict the text feature F_{text}, and vice versa. We do the training of CROM-AE on unlabeled data only.

Stage 2 - Supervised fine-tuning: We design a new model for learning the classification task on labeled data. Firstly, pre-trained encoders of CROM-AE are frozen to extract latent representations from original CLIP features. Then both latent representations (cooked features) and original CLIP features (raw features) were fused to predict the classification target. We call the model Raw and Cooked Features Classification Model (RAW-N-COOK).

4.2 Cross Modality Auto Encoder (CROM-AE)

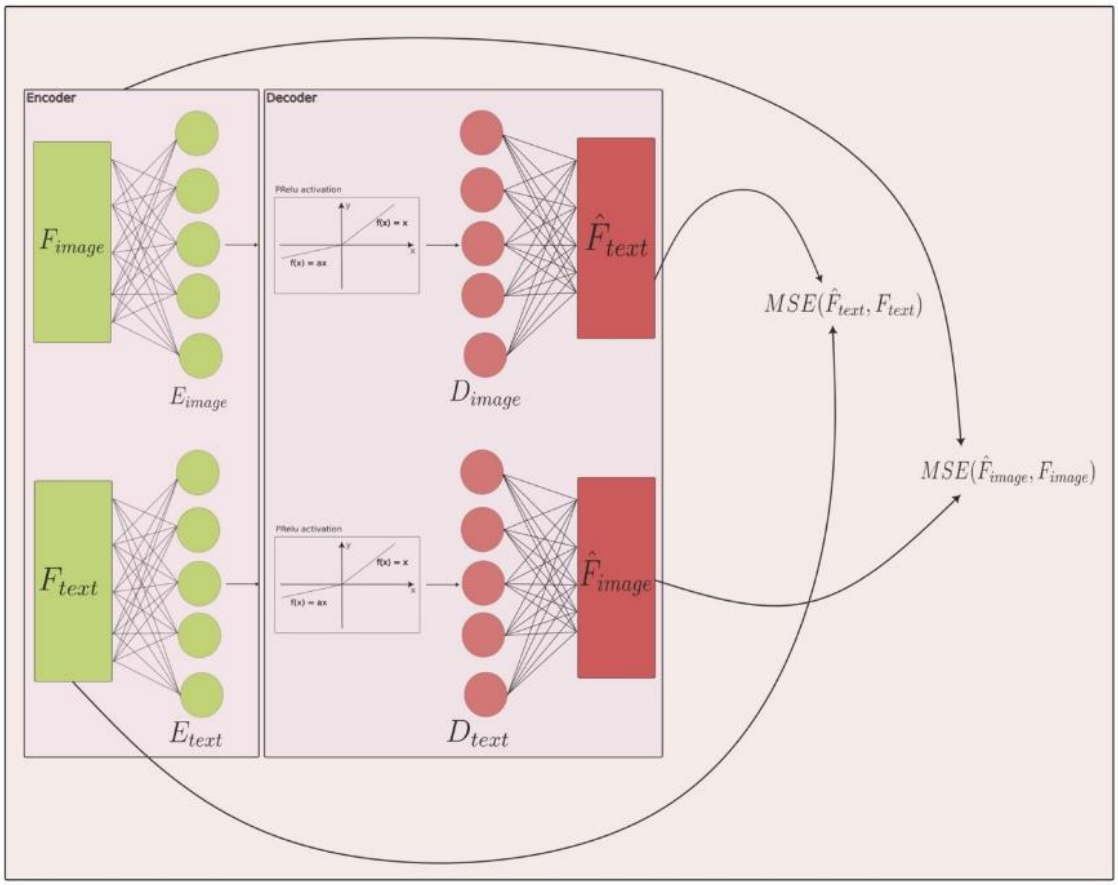

Figure 19. The pipeline of two CROM-AEs

We defined Cross Modality Auto Encoder (CROM-AE) as a model that uses one modality to reconstruct the other modality. We designed two CROM-AE models, AE_{image} and AE_{text}. AE_{image} received CLIP features F_{image} and take F_{text} as the target. AE_{text} do the same, but with the pair input and target is F_{text} and F_{image}. Formally, we have the following:

$$\hat{F}_{text} = AE_{image}(F_{image}) \quad (17)$$

$$\hat{F}_{image} = AE_{text}(F_{text}) \quad (18)$$

where F_{image}, F_{text} are CLIP features of image and text, and $\hat{F}_{text}$, $\hat{F}_{image}$ are estimations of AE_{image} and AE_{text}, respectively.

In practice, our two CROM-AE models have the same simple underlying architecture: Input > Linear > PReLU > Linear > Output. In each CROM-AE model, the encoder includes the first two layers: Input, Linear, while the decoder includes the others. All layers have the same size

of 768. Instead of the popular activation function ReLU, we use PReLU on the encoder's output to force the CROM-AE model to learn the meaningful negative values of the latent representations, which will be helpful in the later phase (*). We denote the encoder linear layer and decoder linear layer of each modality as E_k, D_k $with$ $k \in \{image, text\}$ in Figure 19.

CROM-AE can be seen as a good way to capture the underlying distribution of each modality for semi-supervised learning. Concretely, the latent representations of images are driven by the remaining text distribution p(text), which might contain the information of posterior distribution p(y|text), where y is the supervised classification target. Similarly, text latent representations are guided by p(image), thereby p(y|image).

Because we do not want to introduce new bias and variance to the labeled training set, validation set, and test set, we exclude all labeled data when training CROM-AE. The two CROM-AE models were trained separately with the Mean-Square-Error loss function:

$$\mathcal{L}_{AE_{image}} = MSE\left(\hat{F}_{text}, F_{text}\right) \quad \textbf{(19)}$$

$$\mathcal{L}_{AE_{text}} = MSE\left(\hat{F}_{image}, F_{image}\right) \quad \textbf{(20)}$$

4.3 Raw and Cooked Features Classification Model (RAW-N-COOK)

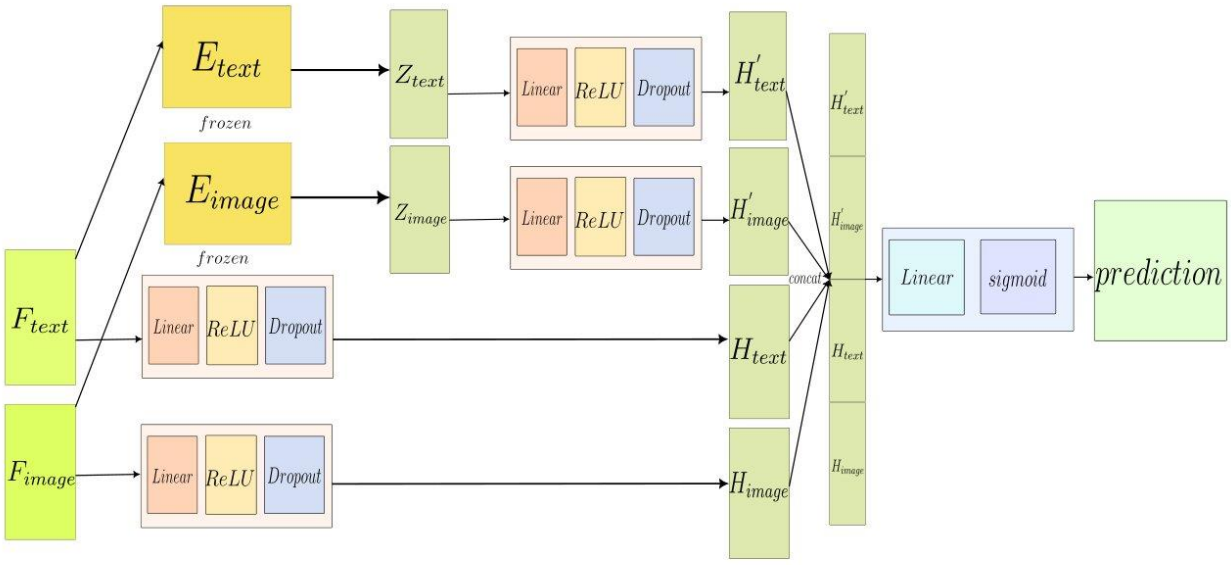

Figure 20. The architecture of our proposed finetune model RAW-N-COOK

RAW-N-COOK is a classification model that incorporates both learned latent representation from CROM-AE and the original CLIP features as follows. Firstly, we take only the encoder part E_{image}, E_{text} of two pre-trained CROM-AE models and freeze them. Then, both CLIP features F_{image} and F_{text} go through their corresponding CROM-AE encoder E_{image}, E_{text} to obtain latent representation Z_{image}, Z_{text}. Then, four vectors: $F_{image}, F_{text}, Z_{image}, Z_{text}$ are projected to four 256-length vectors by a simple sequence of layers: Linear > ReLU > Dropout, then concatenate to obtain a 1024-length vector. The concatenated vector goes through the last Linear layer to learn the classification target. The flow is described in Figure 20.

Our intuition is that Z_{image}, and Z_{text} are informative features because they were learned on a large unlabeled dataset. Therefore, encoders are frozen to keep what CROM-AE learned on unlabeled data. However, both Z_{image} and Z_{text} were driven by different tasks, if we use only Z_{image} and Z_{text} to classification, it is not too powerful. Therefore, we decided to fuse Z_{image} and Z_{text} (cooked features) with the original features outputted from CLIP F_{image}, F_{text} (raw features).

Recall (*), if we use ReLU in the decoder, different negative values on Z_{image}, and Z_{text} are not learned by CROM-AE that will become noise in the classification model, which makes the model harder to learn on the classification task. Therefore, in CROM-AE, we choose PReLU.

Due to the multi-label and class imbalance in our dataset, we train the classification model with Focal Distribution Balanced Loss (DB Loss). A detail of the loss can be seen in 2.7.

Chapter 5 Experiments

5.1 Evaluation metric

5.1.1 Weighted F1 Score

We use weighted- F1 score as the measurement metric which is also the official metric used in MAMI competition [9] for subtask B. The F1 metric is the weighted average sum of macro F1 on each label where the weight is the label's support. First of all, F1 will be calculated for each class:

$$F1(class = 1) = \frac{TP}{TP + \frac{1}{2}(FP + FN)} \quad (21)$$

$$F1(class = 0) = \frac{TN}{TN + \frac{1}{2}(FP + FN)} \quad (22)$$

Secondly, the macro F1 is defined as the average of the classes' F1 scores:

$$F1_{macro} = \frac{F1(class = 0) + F1(class = 1)}{2} \quad (23)$$

Finally, each label's F1 macro will be summed, followed by their supports:

$$F1_{weighted} = \frac{1}{\sum_i support_i} \sum_{i=1}^{n} F1_{macro_i} \cdot support_i \quad (24)$$

The F1 macro score considers both positive and negative classes, thus making it robust for imbalanced class datasets. Furthermore, F1-weighted is calculated based on the number of labels' supports, so each label's F1 macro score can be considered more fairly when imbalance phenomena exist.

5.1.2 AUC score

To know the definition of the AUC score, we need to explain the receiver operating characteristic curve (**ROC curve**). The ROC curve plots the true positive rate (TPR) against the false positive rate (FPR) at different classification thresholds, as shown in Figure 21.
TPR can be written as:

$$TPR = \frac{TP}{TP + FN} \quad (25)$$

Where TP, FN are the number of true positives and false negatives.
FPR is:

$$FPR = \frac{FP}{FP + TN} \quad (26)$$

Where FP, TN are the number of false positives and true negatives.

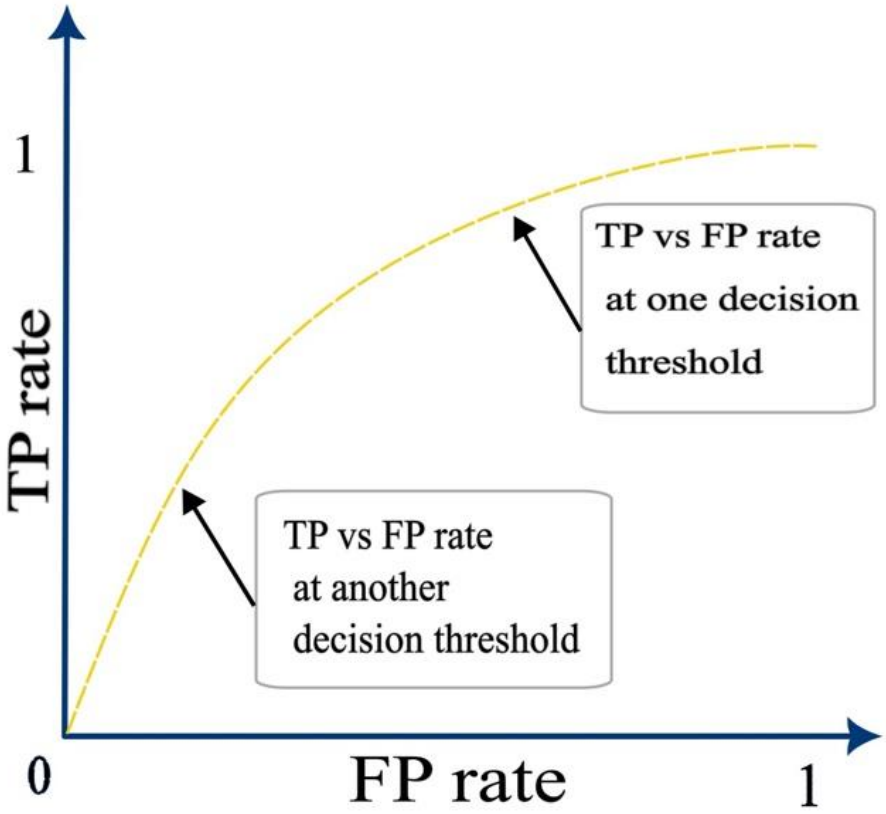

Figure 21. Illustration of ROC curve

AUC is an abbreviation for the area under the ROC curve. AUC measures the likelihood that true positive samples are ranked higher than true negative samples by the magnitude of the area under the ROC curve at all classification thresholds.

5.2 Experiment setting

We choose 3 labeled data ratio setup p = {0.05,0.1,0.3}. For hyperparameter settings, we set the initial learning rate, batch size, number of supervised epochs, number of pre-trained epochs, and random seed to be 1e-4, 40, 50, 100, and 42, respectively. We use CLIP's ViT-L/14 which uses ViT-L/14 transformer architecture for the image encoder and masked self-attention for the text encoder, we set a length of 77 for CLIP's max sequence length. The dropout rate is set to 0.2. We use a single-step scheduler and Adam optimizer. To have optimal performance, we set a suitable γ decay rate after one epoch for each ratio as shown in Table 2.

Table 2. Hyperparameters gamma for different settings

Ratio	γ (StepLR)	Epochs	Learning rate	Batch size	Optimizer
0.3	0.85	50	1e-4	40	Adam
0.1	0.9	50	1e-4	40	Adam
0.05	0.93	50	1e-4	40	Adam

5.3 Benchmark

We benchmark our method with multimodal semi-supervised, self-supervised, and fully supervised methods. We use original CMML and CLIP-integrated CMML for semi-supervised methods; Visual Bert [23], Lxmert [41], Expienet [39], and Vilt [21] for self-supervised methods; and the top solution on MAMI leaderboard TIB-VA [14] for supervised methods.

CLIP-CMML: follows the original CMML framework, but we replace the modality's backbones with CLIP encoders. During training, we iterate 2 separate data loaders for supervised and unsupervised batches at the same time, which can make a stochastic combination of supervised and unsupervised samples, this is different from the original CMML which uses a single data loader resulting in the same supervised and unsupervised samples in one batch across epochs.

Visual Bert: is a pre-trained visual linguistic model released in 2019. It is an early fusion multimodal transformer model. Visual Bert uses region feature bounding boxes as image input features.

Lxmert: is a pre-trained cross-modal self-attention model. The model uses bounding box region features and bounding box information such as coordinates as image input features.

Vilt: focuses on reducing the complexity of visual embedder while also achieving competitive results with other visual-linguistic models using region features, Vilt uses patch projection to produce image input features.

Expienet: is a self-supervised method pre-trained on large-scale in-domain data such as MHSK150K and hateful memes. Expienet uses a contrastive framework for pre-training and achieves a high F1 score on the Memotion datasets.

TIB-VA: is the top winner solution in the MAMI competition. TIB-VA uses CLIP encoders as backbones and training multitasking for both subtask A, and subtask B. We run TIB-VA with the original multitasking setting and measure scores of 4 labels in subtask B as we find that the score of task B is improved when training with multitasking compared with training only task B in TIB-VA.

For self-supervised models such as ViLT, Lxmert, and Visual Bert, we use pre-trained weights from hugging-face and attach a linear classification layer to classify the model's output. For Extpienet, we use the pre-trained weights that the authors have provided and perform classification as the original setup. We train semi-supervised methods with both unlabeled and labeled datasets. We measure the weighted F1 score on the test set after training 50 epochs.

Table 3. Weighted-average F1-Measure on Test Set

Ratio	Ours	CMML origin	CMML-CLIP	TIB-VA	Visual BERT	Extpienet	ViLT	Lxmert
0.3	**0.7433**	0.5673	0.7265	0.7083	0.6337	0.4422	0.6010	0.6005
0.1	**0.7184**	0.5496	0.6872	0.5885	0.5583	0.4139	0.5566	0.5597
0.05	**0.6792**	0.5256	0.6496	0.4150	0.5174	0.4139	0.5456	0.5303

As can be seen in Table 3, our model performs robustly on all three ratio settings. With 30% of label data, we have an F1 score better than CLIP-CMML with a margin of 0.0168 and the leaderboard supervision solution with 0.0350. In 0.05 ratio settings, the supervised model TIB-VA struggles to learn with the test score of only 0.415 while our model still manages to achieve 0.6792.

Table 4. AUC Measure on Test Set

Ratio	Ours	CMML origin	CMML-CLIP	TIB-VA	Visual BERT	Extpienet	ViLT	Lxmert
0.3	**0.8310**	0.6234	0.8289	0.8333	0.6825	0.5043	0.6948	0.6684
0.1	**0.8145**	0.5794	0.7956	0.7901	0.6119	0.4833	0.6312	0.6163
0.05	**0.7989**	0.5647	0.7664	0.7093	0.5604	0.4761	0.5978	0.5986

Table 4 shows our benchmark AUC scores on different label ratio settings. In 5% and 10% label settings, our model achieves the highest scores. While in the 30% label setting, TIB-VA achieves a higher score than us with a difference of 0.0023. Although TIB-VA has a higher AUC score, its F1 weighted score is still lower than ours. This can be deduced that the predictions of TIB-VA still contain many false negative samples while our method has purposefully tried to reduce that.

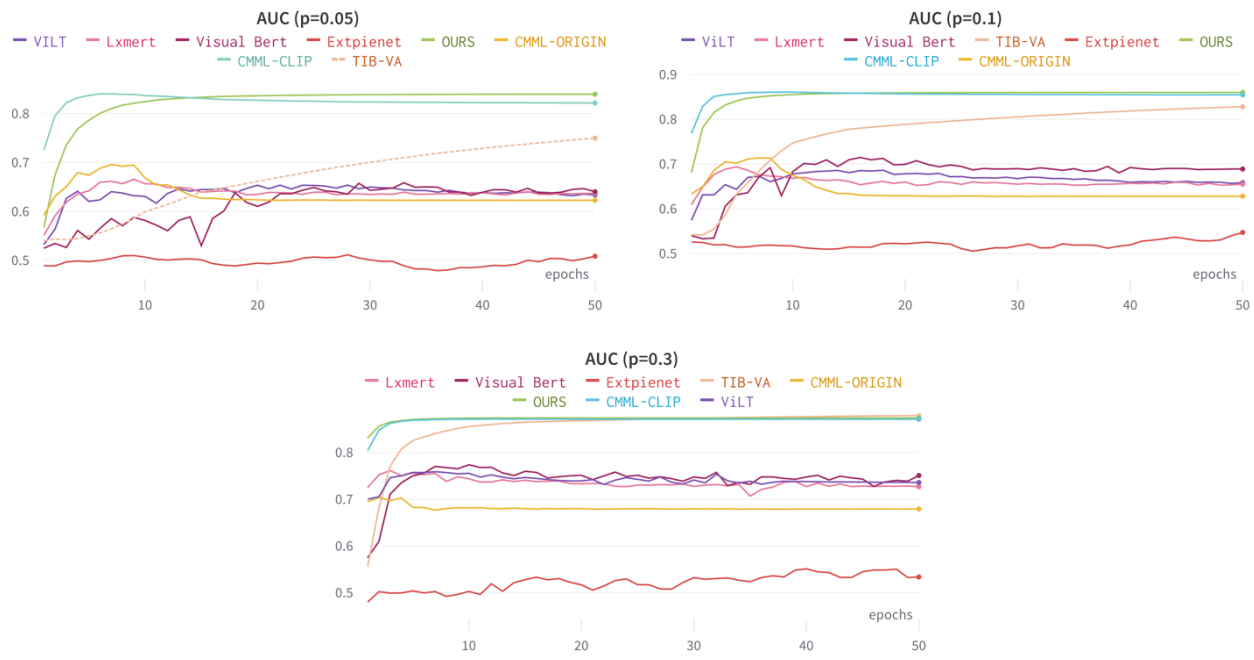

Figure 22. AUC on validation set benchmarked on 8 methods with 50 epochs training

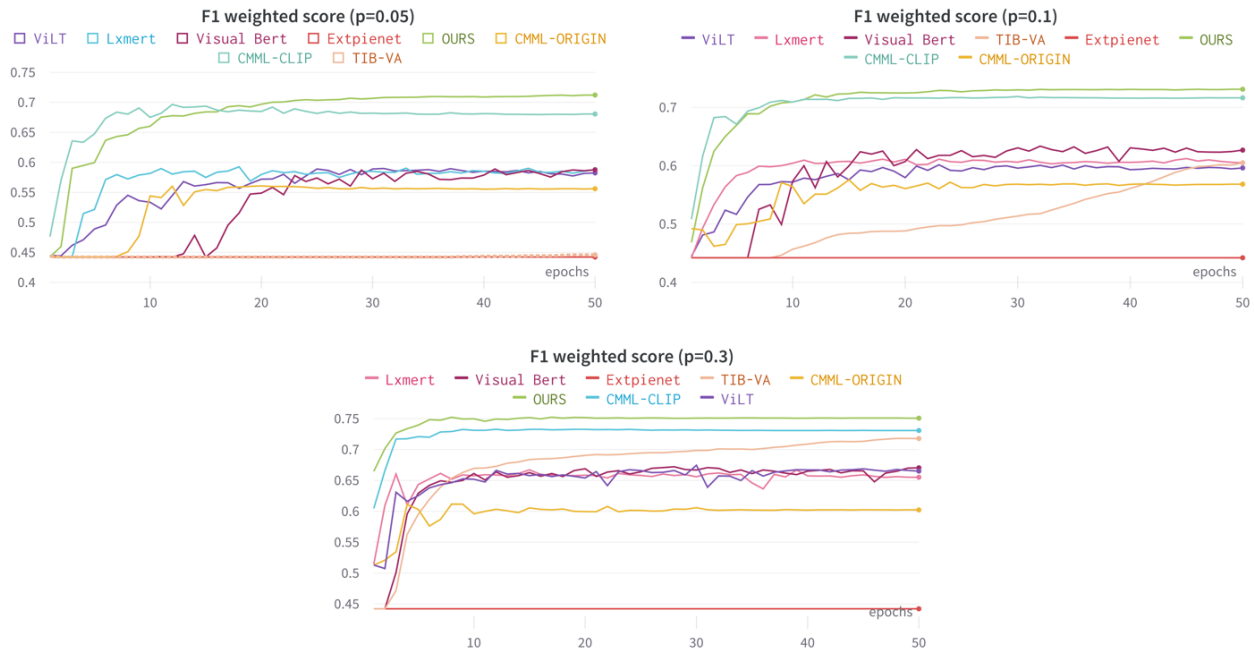

Figure 23. F1 weighted on validation set benchmarked on 8 methods with 50 epochs

Figure 22 and Figure 23 show the AUC and F1 weighted scores of all benchmark models on the validation set with 50 epochs. First of all, the convergence of our method is quite fast, usually in the 10th epoch, and always greater than 0.7 F1 and 0.8 for AUC. Our model's performances surpass other pre-trained visual linguistic models like Visual Bert, Lxmert,

Vilt, Extpienet, and semi-supervised model CMML-Origin in low-labeled data settings. However, the performance of CMML-CLIP comes close to ours, but it does not exceed our scores for all settings. TIB-VA's validation scores are lower than ours in the 5% and 10% labeled setting, but in the 30% labeled setting, the AUC of TIB-VA is greater than our model with a margin of 0.005 while its F1 is still lower with the difference of 0.033. Interestingly, in the 5% labeled setting, TIB-VA's score is quite low (0.4534 F1), it improves dramatically when the labeled setting increases (0.605 at 5% labeled ratio and 0.7178 at 30% labeled ratio) which can be attributed to the label efficiency property of CLIP backbone in TIB-VA needed to have at least a number of labeled samples to have a competitive performance.

5.4 Analysis

To investigate the mechanism of our CROM-AE encoders, we plot the 2d-projection of 4 types of features F_{image}, F_{text}, Z_{image}, and Z_{text}, where F_{image}, F_{text} are the outputs from CLIP, $Z_{image} = E_{image}(F_{image})$, $Z_{text} = E_{text}(F_{text})$ are the output of our encoders E_{image} and E_{text} after training CROM-AE. As presented, F_{image}, F_{text}, Z_{image}, and Z_{text} are vectors that will be fused to find the target in the fine-tuning phase. We do the analysis for the setup of p = 0.1 labeled data on both labeled and unlabeled data. The projection is done by Principal Component Analysis (PCA). For convenience, on labeled data, if at least 1 of 4 labels of a sample are equal to 1, we annotate the sample as "Misogynous" in visualization, otherwise, we annotate it as "Not Misogynous", which is also equivalent to the task A of MAMI dataset.

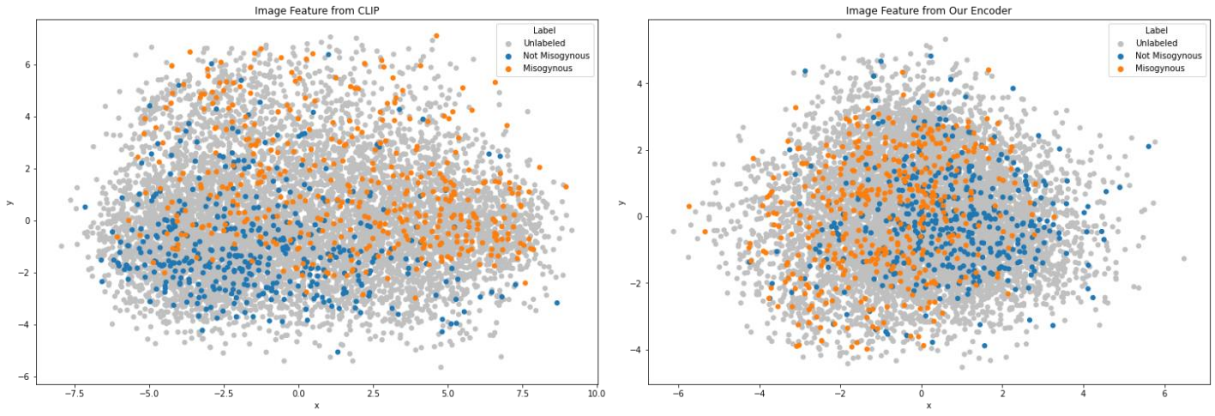

Figure 24. The PCA projection of F_{image} and Z_{image} from left to right

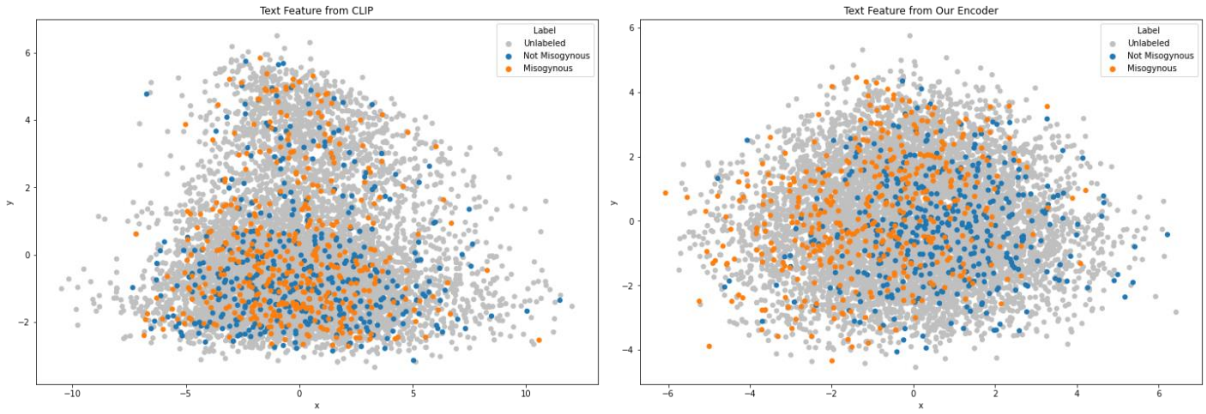

Figure 25. The PCA projection of F_{text} and Z_{text} from left to right

As we can see in Figure 24, the CLIP feature F_{image} is quite good when it tends to distribute in different zones for different classes without training in the downstream task. Although Z_{image} is not improved so much from F_{image}, still tries to keep the disentanglement of classes. About text features in Figure 25, even without seeing labels when training, the encoder's output Z_{text} disentangles text CLIP feature F_{text} a lot. The text encoder is excellent for mapping data within a class to the same region without requiring labels.

Because all four types of features F_{image}, F_{text}, Z_{image}, and Z_{text} are helpful, ensemble them in a fusion model in the fine-tuning stages is a reasonable way.

5.5 Ablation study

Table 5. Ablation: Weighted-average F1-Measure on Test Set

Ratio	Ours	w/o CROM-AE encoders	w/o DB loss	w/o DB loss + CROM-AE encoders
0.3	0.7433	0.7365	0.7316	0.7074

We further perform an ablation study to inspect which components of our proposed method most affect the result. We experiment with 4 modes. Firstly, we train our best model. Secondly, we remove the pre-trained CROM-AE encoders. Thirdly, we remove DB loss. Finally, we remove both DB loss and pre-trained CROM-AE encoders. All the hyperparameter is still kept the same, and we train in 50 epochs. In the setting without DB loss, we replace DB loss with BCE loss. In the setting without CROM-AE encoders, we removed E_{image}, E_{text} and their output's projected 256-length vectors, i.e., concatenate projected vectors of CLIP features only (see Figure 20). As shown in Table 5, without the CROM-AE encoders, the F1 score decreases by 0.0068, and without DB loss, the F1 score drops a bit greater about 0.0117. Finally, in the setting without the DB loss and CROM-AE encoders, the model drops

significantly to 0.7074 which is near TIB-VA's performance (0.7083). Overall, it can be said that the DB loss and CROM-AE encoders contribute equally to achieving our final result.

Chapter 6 Conclusion and Future works

In summary, we successfully applied semi-supervised learning for sentiment analysis of memes in the image macros form. We created an approach surpassing current SOTA multimodal semi-supervised learning and supervised learning methods on the same amount of labeled data. Besides employing CLIP features, our approach consists of multiple improvements: unsupervised pre-train on unlabeled data, integrating pre-train models into the supervised model, and applying a balance loss function to deal with the class imbalance problem on labeled data for the supervised model. Our approach can help to leverage a huge amount of unlabeled memes on the internet and solve the annotation process's pain points.

Due to multiple discrete stages, our approach is quite complicated to execute. It also depends on features CLIP model extracted and being constrained by frozen layers, limiting the ceil our approach can reach. Therefore, we suggest an end-to-end approach to train the entire flow, including CLIP backbone, appropriate unsupervised tasks, and supervised tasks, in a unified framework for future work.

Reference

[1] Börzsei, L.K. 2013. Makes a meme instead. *The Selected Works of Linda Börzsei*. (2013), 1–28.

[2] Chua, T.-S., Tang, J., Hong, R., Li, H., Luo, Z. and Zheng, Y. 2009. NUS-WIDE: A Real-World Web Image Database from National University of Singapore. *Proceedings of the ACM International Conference on Image and Video Retrieval* (New York, NY, USA, 2009).

[3] Clark, K., Luong, M.-T., Le, Q.V. and Manning, C.D. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*. (2020).

[4] Devlin, J., Chang, M.-W., Lee, K. and Toutanova, K. 1810. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv 2018. *arXiv preprint arXiv:1810.04805*. (1810), 0–85083815650.

[5] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., and others 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*. (2020).

[6] van Engelen, J.E. and Hoos, H.H. 2020. A survey on semi-supervised learning. *Machine Learning*. 109, 2 (Feb. 2020), 373–440. DOI:https://doi.org/10.1007/s10994-019-05855-6.

[7] Erhan, D., Courville, A., Bengio, Y. and Vincent, P. 2010. Why does unsupervised pre-training help deep learning? *Proceedings of the thirteenth international conference on artificial intelligence and statistics* (2010), 201–208.

[8] Escalante, H.J., Hernández, C.A., Gonzalez, J.A., López-López, A., Montes, M., Morales, E.F., Enrique Sucar, L., Villaseñor, L. and Grubinger, M. 2010. The Segmented and Annotated IAPR TC-12 Benchmark. *Comput. Vis. Image Underst.* 114, 4 (Apr. 2010), 419–428. DOI:https://doi.org/10.1016/j.cviu.2009.03.008.

[9] Fersini, E., Gasparini, F., Rizzi, G., Saibene, A., Chulvi, B., Rosso, P., Lees, A. and Sorensen, J. 2022. SemEval-2022 Task 5: Multimedia Automatic Misogyny Identification. *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)* (Seattle, United States, Jul. 2022), 533–549.

[10] Goldberg, A., Zhu, X., Singh, A., Xu, Z. and Nowak, R. 2009. Multi-Manifold Semi-Supervised Learning. *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics* (Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA, Apr. 2009), 169–176.

[11] Gomez, R., Gibert, J., Gomez, L. and Karatzas, D. 2020. Exploring hate speech detection in multimodal publications. *Proceedings of the IEEE/CVF winter conference on applications of computer vision* (2020), 1470–1478.

[12] Goodfellow, I., Bengio, Y. and Courville, A. 2016. *Deep Learning*. MIT Press.

[13] Gunti, N., Ramamoorthy, S., Patwa, P. and Das, A. 2022. Memotion Analysis through the Lens of Joint Embedding (Student Abstract). (2022).

[14] Hakimov, S., Cheema, G.S. and Ewerth, R. 2022. TIB-VA at SemEval-2022 Task 5: A Multimodal Architecture for the Detection and Classification of Misogynous Memes. *arXiv preprint arXiv:2204.06299*. (2022).

[15] He, K., Zhang, X., Ren, S. and Sun, J. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *Proceedings of the IEEE international conference on computer vision* (2015), 1026–1034.
[16] He, P., Liu, X., Gao, J. and Chen, W. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*. (2020).
[17] Hu, P., Zhu, H., Peng, X. and Lin, J. 2020. Semi-supervised multi-modal learning with balanced spectral decomposition. *Proceedings of the AAAI Conference on Artificial Intelligence* (2020), 99–106.
[18] Huiskes, M.J. and Lew, M.S. 2008. The MIR Flickr Retrieval Evaluation. *Proceedings of the 1st ACM International Conference on Multimedia Information Retrieval* (New York, NY, USA, 2008), 39–43.
[19] Kiela, D. et al. 2021. The Hateful Memes Challenge: Competition Report. *Proceedings of the NeurIPS 2020 Competition and Demonstration Track* (Dec. 2021), 344–360.
[20] Kiela, D., Firooz, H., Mohan, A., Goswami, V., Singh, A., Ringshia, P. and Testuggine, D. 2020. The hateful memes challenge: Detecting hate speech in multimodal memes. *Advances in Neural Information Processing Systems*. 33, (2020), 2611–2624.
[21] Kim, W., Son, B. and Kim, I. 2021. Vilt: Vision-and-language transformer without convolution or region supervision. *International Conference on Machine Learning* (2021), 5583–5594.
[22] Kumar, G.K. and Nanadakumar, K. 2022. Hate-CLIPper: Multimodal Hateful Meme Classification based on Cross-modal Interaction of CLIP Features. *arXiv preprint arXiv:2210.05916*. (2022).
[23] Li, L.H., Yatskar, M., Yin, D., Hsieh, C.-J. and Chang, K.-W. 2019. VisualBERT: A Simple and Performant Baseline for Vision and Language. *CoRR*. abs/1908.03557, (2019).
[24] Liang, J., Li, R. and Jin, Q. 2020. Semi-supervised multi-modal emotion recognition with cross-modal distribution matching. *Proceedings of the 28th ACM International Conference on Multimedia* (2020), 2852–2861.
[25] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P. and Zitnick, C.L. 2014. Microsoft COCO: Common Objects in Context. *Computer Vision – ECCV 2014* (Cham, 2014), 740–755.
[26] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L. and Stoyanov, V. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*. (2019).
[27] Lu, L. 2020. Dying ReLU and Initialization: Theory and Numerical Examples. *Communications in Computational Physics*. 28, 5 (2020), 1671–1706.
[28] Lugea, J. 2019. The pragma-stylistics of 'image macro'internet memes. *Contemporary media stylistics*. (2019), 81–106.
[29] Ouali, Y., Hudelot, C. and Tami, M. 2020. An Overview of Deep Semi-Supervised Learning. *CoRR*. abs/2006.05278, (2020).
[30] Patwa, P., Ramamoorthy, S., Gunti, N., Mishra, S., Suryavardan, S., Reganti, A., Das, A., Chakraborty, T., Sheth, A., Ekbal, A., and others 2021. Findings of Memotion 2: Sentiment and Emotion Analysis of Memes. (2021).
[31] Pramanick, S., Dimitrov, D., Mukherjee, R., Sharma, S., Akhtar, M.S., Nakov, P. and Chakraborty, T. 2021. Detecting Harmful Memes and Their Targets. *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021* (2021), 2783–2796.

[32] Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., and others 2021. Learning transferable visual models from natural language supervision. *International Conference on Machine Learning* (2021), 8748–8763.
[33] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., and others 2019. Language models are unsupervised multitask learners. *OpenAI blog*. 1, 8 (2019), 9.
[34] Ramamoorthy, S., Gunti, N., Mishra, S., Suryavardan, S., Reganti, A., Patwa, P., Das, A., Chakraborty, T., Sheth, A., Ekbal, A., and others 2021. Memotion 2: Dataset on Sentiment and Emotion Analysis of Memes. (2021).
[35] Ridnik, T., Ben-Baruch, E., Zamir, N., Noy, A., Friedman, I., Protter, M. and Zelnik-Manor, L. 2021. Asymmetric loss for multi-label classification. *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2021), 82–91.
[36] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., and others 2015. Imagenet large scale visual recognition challenge. *International journal of computer vision*. 115, 3 (2015), 211–252.
[37] Sanh, V., Debut, L., Chaumond, J. and Wolf, T. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*. (2019).
[38] Sharma, C., Bhageria, D., Scott, W., Pykl, S., Das, A., Chakraborty, T., Pulabaigari, V. and Gambäck, B. 2020. SemEval-2020 Task 8: Memotion Analysis-the Visuo-Lingual Metaphor! *Proceedings of the Fourteenth Workshop on Semantic Evaluation* (2020), 759–773.
[39] Sharma, S., Siddiqui, M.K., Akhtar, Md.S. and Chakraborty, T. 2022. Domain-aware Self-supervised Pre-training for Label-Efficient Meme Analysis. *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)* (Online only, Nov. 2022), 792–805.
[40] Sunkara, M., Ronanki, S., Bekal, D., Bodapati, S. and Kirchhoff, K. 2020. Multimodal semi-supervised learning framework for punctuation prediction in conversational speech. *arXiv preprint arXiv:2008.00702*. (2020).
[41] Tan, H. and Bansal, M. 2019. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*. (2019).
[42] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, \Lukasz and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*. 30, (2017).
[43] Wu, T., Huang, Q., Liu, Z., Wang, Y. and Lin, D. 2020. Distribution-balanced loss for multi-label classification in long-tailed datasets. *European Conference on Computer Vision* (2020), 162–178.
[44] Yang, X., Song, Z., King, I. and Xu, Z. 2021. A survey on deep semi-supervised learning. *arXiv preprint arXiv:2103.00550*. (2021).
[45] Yang, Y., Wang, K.-T., Zhan, D.-C., Xiong, H. and Jiang, Y. 2019. Comprehensive Semi-Supervised Multi-Modal Learning. *IJCAI* (2019), 4092–4098.
[46] Zhang, Y., Jiang, H., Miura, Y., Manning, C.D. and Langlotz, C.P. 2020. Contrastive learning of medical visual representations from paired images and text. *arXiv preprint arXiv:2010.00747*. (2020).
[47] Zhu, X.J. 2005. Semi-supervised learning literature survey. (2005).