

Representation Learning in Video and Text - A Social Media Misinformation Perspective

*Kehan Wang
Avideh Zakhor, Ed.*



Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2022-140

<http://www2.eecs.berkeley.edu/Pubs/TechRpts/2022/EECS-2022-140.html>

May 18, 2022

Copyright © 2022, by the author(s).
All rights reserved.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission.

Representation Learning in Video and Text - A Social Media Misinformation Perspective


by Kehan Wang

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences,
University of California at Berkeley, in partial satisfaction of the requirements for the
degree of **Master of Science, Plan II**.

Approval for the Report and Comprehensive Examination:

Committee:


Avideh ZAKHOR (May 18, 2022 16:43 PDT)

Professor Avideh Zakhor
Research Advisor

May 18, 2022

(Date)

* * * * *



Professor Alexei Efros
Second Reader

May 18, 2022

(Date)

Representation Learning in Video and Text - A Social Media Misinformation Perspective

by

Kehan Wang

A thesis submitted in partial satisfaction of the
requirements for the degree of

Master of Science

in

Electrical Engineering and Computer Sciences

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Avidesh Zakhori, Chair
Professor Alexei Efros

Spring 2022

Abstract

Representation Learning in Video and Text - A Social Media Misinformation Perspective

by

Kehan Wang

Master of Science in Electrical Engineering and Computer Sciences

University of California, Berkeley

Professor Avidoh Zakhori, Chair

Short videos have become the most popular form of social media in recent years. As popular short videos easily gather billions of views, recent studies find that misinformation spreads faster through videos, and viewers have trouble identifying misinformation in social media. In this work, we develop self-supervised and unsupervised methods to identify misinformation by detecting inconsistency across multiple modalities, namely video and text. We explore both contrastive learning and masked language modeling (MLM) on a dataset of one million Twitter posts spanning from 2021 to 2022. Our best performing method outperforms state-of-the-art methods by over 9% in accuracy. We further show that the performance of random mismatch detection transfers on actual misinformation on a manually-labeled dataset of 401 posts. For this dataset, our method outperforms state-of-the-art methods by over 14% in accuracy.

To my parents, my girlfriend, and my friends

I am eternally grateful for my parents' unconditional support, thankful for my girlfriend's love and remote accompany, and lucky to have my friends who always make me happy.

Thank you.

Contents

Contents	ii
List of Figures	iv
List of Tables	vi
1 Introduction	1
2 Related Work	5
2.1 Video and Language Understanding	5
2.2 Misinformation Detection	7
3 Video-text Misinformation Detection Pipeline	9
4 Contrastive Learning for Misinformation Detection	11
4.1 Background on Transformer	11
4.2 Feature Aggregation	12
4.3 Contrastive Learning	13
4.3.1 Contrastive Loss	13
4.3.2 Noise Contrastive Estimation Loss	14
4.4 Video-text Fusion Methods	16
4.4.1 Direct Concatenation(DC)	16
4.4.2 Cross-Transformer(CT)	17
4.5 Evaluations	18
4.5.1 Contrastive Learning	18
4.5.2 Comparison of Fusion Methods	19
5 Masked Language Modeling for Misinformation Detection	21
5.1 Masked Language Modeling	21
5.2 Learned Representation Space	23
5.2.1 Global Mean Pooling(GMP)	23
5.2.2 [CLS] Extraction through Attention(CLS)	24
5.3 Evaluations	24

5.3.1	MLM Loss	24
5.3.2	Comparison of Representation Extraction Methods	25
5.4	Unsupervised Misinformation Grounding	25
6	Experiments and Ablation Studies	29
6.1	Data Collection and Filtering	29
6.2	Performance Comparison	30
6.3	Qualitative Examples	31
6.4	Ablation Studies	32
6.4.1	Text Feature Extractor	33
6.4.2	Effect of Two-stage Training	33
6.5	Performance Comparison on Topic-Specific Random Mismatch	35
7	Conclusion and Future Work	37
7.1	Conclusion	37
7.2	Future Work	37
7.3	Engineering Traps to Avoid in Vision-and-Language Research	38
	Bibliography	39

List of Figures

1.1	Popular Scrolling-Feed Short-Video Social Media Platforms – TikTok, Twitter, Youtube Shorts, Instagram Reels.	1
1.2	Edited video in Context of the Russia-Ukraine Crisis – (a) Original video on the left, showing Ukraine President Zelensky at a video conference; (b) Edited video on the right, with a pile of cocaine on the table to defame him. See the video here.	2
1.3	Example of misinformation in a social media video post – the video’s mismatched text description in contains activity mismatch and topical shift. . .	3
2.1	Methods of Fusing Video and Text Features – (a) Direct Concatenation; (b) Extending BERT to include video; (c) Cross-encoder.	6
3.1	Misinformation Detection Pipeline	9
3.2	Feature Extraction Using S3D and DeBERTa-v3	10
4.1	Transformer Layer Architecture	12
4.2	Project Video and Text Features to the Same Dimension	13
4.3	Contrastive Learning – separate transformers for each modality, aggregation through mean pooling, contrastive learning on aggregated features.	14
4.4	Projection of Video and Text Representations for NCE Loss.	15
4.5	Fusion Methods in Using Learned Representation Space – (a) Direct Concatenation(DC); (b) Cross-Transformer(CT).	17
5.1	Masked Language Modeling – Model Architecture	21
5.2	Methods of Extracting Representation from Learned Representation space – (a) Global Mean Pooling; (b) Attention.	23
5.3	Grounding Misinformation using Unsupervised Learning – (a) Matching video; (b) Mismatching video. Each word is evaluated on whether it brings in misinformation to video as a description.	26
6.1	Video and Text Length Distribution in Collected Twitter Dataset after Filtering. – (a) Number of characters in Text; (b) Length of videos in seconds	29
6.2	Misinformation Labeling System for Twitter Video Posts	30

6.3	Examples of Our Model’s Correct Predictions in Labeled Misinformation Dataset – (a) Matched post; (b) Mismatched post.	32
6.4	Examples of Our Model’s Wrong Predictions in Labeled Misinformation Dataset – (a) Matched post; (b) Mismatched post.	32

List of Tables

4.1	Effect of Contrastive Learning on Random Mismatch Detection – NCE loss improves accuracy by 4.58%, compared to only using BCE loss to learn classification.	18
4.2	Effect of Contrastive Learning on Labeled Misinformation Detection – NCE loss improves accuracy by 0.75%, compared to only using BCE loss to learn classification.	19
4.3	Effect of Fusion Method on Random Mismatch Detection – Cross-Encoder performs 1.07% higher than direct concatenation.	19
4.4	Effect of Fusion Method on Labeled Misinformation Detection – Cross-Encoder performs 1.24% higher than direct concatenation.	20
5.1	Effect of Masked Language Modeling on Random Mismatch Detection – MLM loss improves accuracy by 1.41%, compared to only using BCE loss to learn classification.	24
5.2	Effect of Masked Language Modeling on Labeled Misinformation Detection – MLM loss improves accuracy by 3.99%, compared to only using BCE loss to learn classification.	25
5.3	Effect of Representation Extraction Method on Random Mismatch Detection – [CLS] token outperforms global mean pooling by 0.9% in accuracy.	25
5.4	Effect of Representation Extraction Method on Labeled Misinformation Detection – [CLS] token outperforms global mean pooling by 4.74% in accuracy.	25
5.5	Inference Methods of Unsupervised Learned Model on Random Mismatch Detection – methods consistently predict more negative than positive, resulting in accuracy of a random guess, 50%.	27
5.6	Inference Methods of Unsupervised Learned Model on Labeled Misinformation Detection – methods consistently predict more negative than positive, resulting in high Precision and M-Recall, but poor accuracy overall.	28
6.1	Performance Comparison on Random Mismatch Detection – Contrastive Learning method outperforms McCrae et al. in misinformation detection accuracy by 9.03%.	31

6.2	Performance Comparison on Labeled Misinformation Detection – Contrastive Learning method outperforms McCrae et al. in misinformation detection accuracy by 14.96%.	31
6.3	Comparison of BERT and DeBERTa – DeBERTa is newer, larger, and performs better on MNLI than BERT.	33
6.4	Effect of Text Feature Extractor Quality on Random Mismatch Detection – DeBERTa improves accuracy by 4.64% over BERT. CT - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$ is used as the model.	33
6.5	Effect of Text Feature Extractor Quality on Labeled Misinformation Detection – DeBERTa improves accuracy by 5.97% over BERT. CT - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$ is used as the model.	33
6.6	Effect of Two-Stage Training on Random Mismatch Detection – Pre-training does not help misinformation detection performance.	34
6.7	Effect of Two-Stage Training on Labeled Misinformation Detection – Pre-training helps contrastive learning on classifying misinformation, with higher M-Precision and higher M-Recall.	34
6.8	Random Mismatch Experiment on COVID-19 Related Tweets – State-of-the-art methods only achieve random-guess accuracy, while our methods greatly exceeds random-guess.	35
6.9	Random Mismatch Experiment on Russia-Ukraine Related Tweets – Pre-training improves model accuracy on unseen test distribution.	35

Acknowledgments

I would like to express my sincere gratitude to my advisor Prof. Avidah Zakhor, for countless opportunities and guidance she has offered me. I would also like to thank David Chan and Seth Zhao for working with me on misinformation detection, for the many suggestions provided and for being there when I am experiencing ups and downs in research. I would like to thank Prof. Alexei Efros for serving as the second reader of my thesis, and his wonderful lectures that led me into research. I also like to thank Google for providing funding for Google Cloud. Without this funding, my research would have been impossible to complete. Last but not least, I would like to thank Github Co-pilot for peer-programming with me and making the process easier.

Chapter 1

Introduction

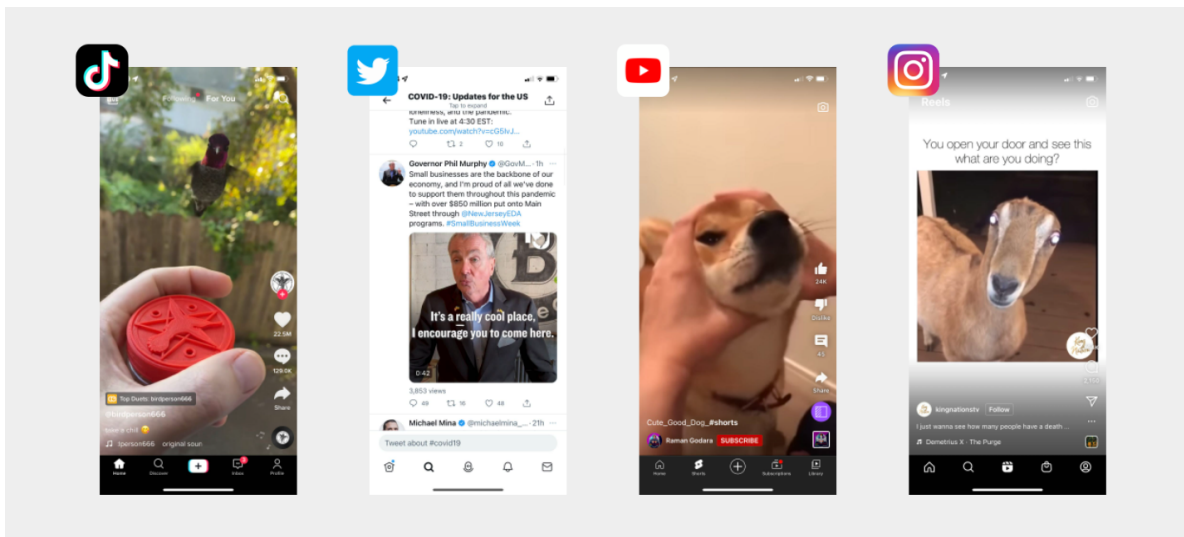


Figure 1.1: Popular Scrolling-Feed Short-Video Social Media Platforms – TikTok, Twitter, Youtube Shorts, Instagram Reels.

Short videos are becoming increasingly popular in social media these days - viral videos on TikTok, Instagram Reels, Twitter and YouTube Shorts, shown in Figure 1.1 are receiving billions of views from all over the world. Often times, short video social media is provided through a scrolling feed view. Within this view, a video is automatically played, and the user can choose to play the next video by scrolling down. Since the currently viewed video is always automatically played, such scrolling feed systems allow users to be highly engaged and constantly consuming new social media posts. The key to the success of these scrolling feed systems lays in its content recommendation algorithm. To make sure users are engaged by the videos in the feed, scrolling feed systems have to understand the video social media posts in order to place videos of users' interest into their feeds. Better content recommendation

leads to higher customer retention rate, which in turn drive up the company’s revenue. This viral trend in scrolling feed systems sparks tremendous efforts in both academia and industry to study how to understand the content of such video posts.

Video social media posts often have one short video as their main component, accompanied by a few sentences as a description or a reaction to the video. Both video and text components are displayed and consumed by users at the same time. In recent years, a great deal of research has been devoted to the study of understanding video and language together in the context of different tasks, such as action localization, video retrieval, video captioning, video question answering and video-text inference.

However, detection of misinformation has received little attention in the context of video-and-language models. Misinformation is false information that is deliberately created to deceive users. It is commonly seen in propaganda and fake news. As photo/video editing techniques become better, it is becoming increasingly more difficult to distinguish between manipulated videos and the original ones.

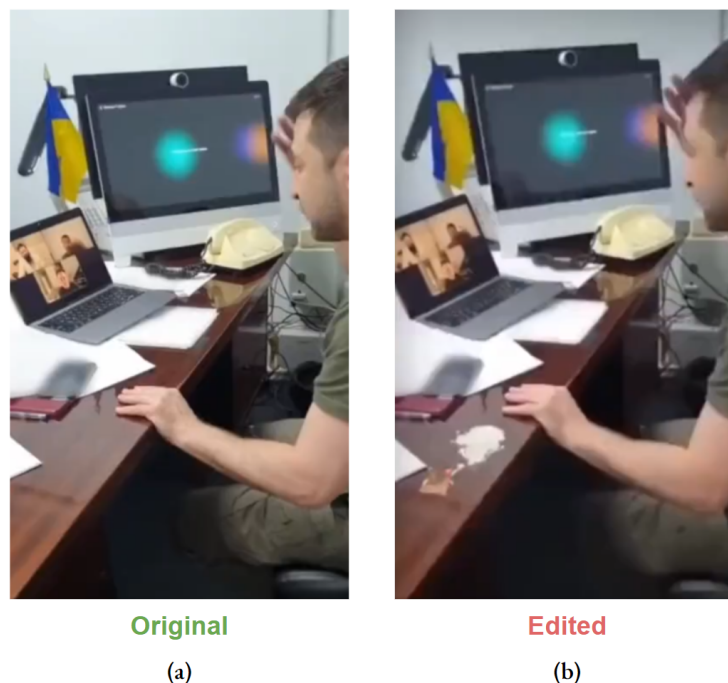


Figure 1.2: **Edited video in Context of the Russia-Ukraine Crisis** – (a) Original video on the left, showing Ukraine President Zelensky at a video conference; (b) Edited video on the right, with a pile of cocaine on the table to defame him. See the video [here](#).

It is now relatively straightforward to create a social media post of misinformation, where the video or text of an original post is edited to convey a different message from the original post. Figure 1.2 shows an example of an edited video. It has been added a pile of cocaine on the table next to President Zelensky to defame him in context of the Russia-Ukraine

Crisis. Throughout recent global events such as the COVID-19 pandemic and the Russia-Ukraine Crisis, much misinformation has spread on social media platforms to deceive the public for political purposes. Indeed, Vosoughi et al. [37] found that the social media posts with falsified information spread faster, and reached a larger audience than posts containing truthful facts. In particular, social media posts of videos spread even faster than posts of only audio and text [32]. Recent studies [3, 24] also found that only 17% Americans can detect fake news better than chance, while 84% claim to be confident in their ability to detect fake news in social media. Misinformation has left 88% confused about basic facts, in times where facts are the most important resources. In the context of the global COVID-19 pandemic, misinformation exposure has caused 6.3% decline in intent to receive vaccination among participants[16]. The harm anti-mask and anti-vaccine campaigns have caused is immeasurable. How can we stop the spread of misinformation?

The first step to fighting misinformation is to detect posts of misinformation. In this thesis, we develop methods to identify posts that contain semantic inconsistencies, where the short video does not semantically match its accompanying description. An example of semantic inconsistency is shown in Figure 1.3.



Matching: COVID-19 vaccines are safe and effective. Get vaccinated today, with just one quick and painless dab!

Mismatching: OMG look what they did to him!! Nobody should put chips in human! He can't breathe anymore!

Figure 1.3: **Example of misinformation in a social media video post** – the video’s mismatched text description in contains activity mismatch and topical shift.

The challenges of misinformation detection in social media video posts are two-fold: (1) learning a joint representation of video and text effectively; (2) lack of a large, labeled dataset for semantic matching. Here, we take steps towards both of these issues. To address the joint representation learning problem, we propose a deep-learning based method for learning accurate video-language joint distributions, and utilize this representation to efficiently detect semantic inconsistencies. To address the data issues, we introduce a novel evaluation method through random-mismatch that does not require extra human labeling effort. We collect one million social media video posts from Twitter to use as a large self-supervised training corpus,

and introduce a novel testing dataset consisting of 401 annotated video social media posts to be used as a gold standard for future unsupervised and self-supervised misinformation detection methods. Our method of representation learning outperforms state-of-the-art methods[20, 30] on random-mismatch detection and the above labeled misinformation dataset by 9.03% and 14.96% respectively. We further show in Section 6.5 that state-of-the-art methods are only detecting topic mismatch in video and text, while our methods are understanding video and text to detect semantic mismatch.

Chapter 2

Related Work

In this chapter, we will describe our proposed methods in the context of existing video-and-language understanding models and multi-media forensics methods.

2.1 Video and Language Understanding

Numerous tasks have been introduced to better understand how models perform on understanding video and language together. Examples of such tasks include action localization, video retrieval, video captioning, video question answering and video-text inference. CrossTask[44] and COIN[34] datasets evaluate action localization within a video using generated ASR descriptions. YouCook[7] and YouCook2[43] are cooking instructional video datasets and are often used to evaluate video retrieval and video captioning. HowTo100M[23] is the largest instructional video dataset containing 136 million video clips and 23,000 activities. It is often used for pre-training video-and-text understanding models to be fine-tuned for downstream tasks. Outside of instructional videos domain, MSR-VTT[41] is a manually labeled dataset for video-to-text retrieval evaluation. TVQA[13], MovieQA[35] and ActivityQA[42] are video question answering datasets of different categories of videos. VIOLIN[15] is a large dataset on video-and-language inference, evaluating model’s video-text understanding through inferring whether claims made about the video-text pair are true. In this work, we propose a similar evaluation method to VIOLIN. Our method does not require extra human-labeling and can measure the effectiveness of video-and-language understanding to the same extent as the above mentioned evaluation tasks.

Video and text are two modalities with temporal information that cannot be easily processed together through one single encoder. Therefore, an important question on video and text joint modeling is, how to fuse their representations together. To address this problem, two different sub-problems have to be addressed: (1) choice of architecture to be used in combining video and language features, and (2) ensuring that the fused video and language features are meaningful.

Regarding fusion architectures, methods proposed nowadays mainly use three different

paradigms, illustrated in Figure 2.1: (1) direct concatenation of video and text features; (2) extending BERT to take in video tokens as input in addition to text; (3) a cross-encoder to jointly process the video and language features without forgoing the temporal information within each modality.

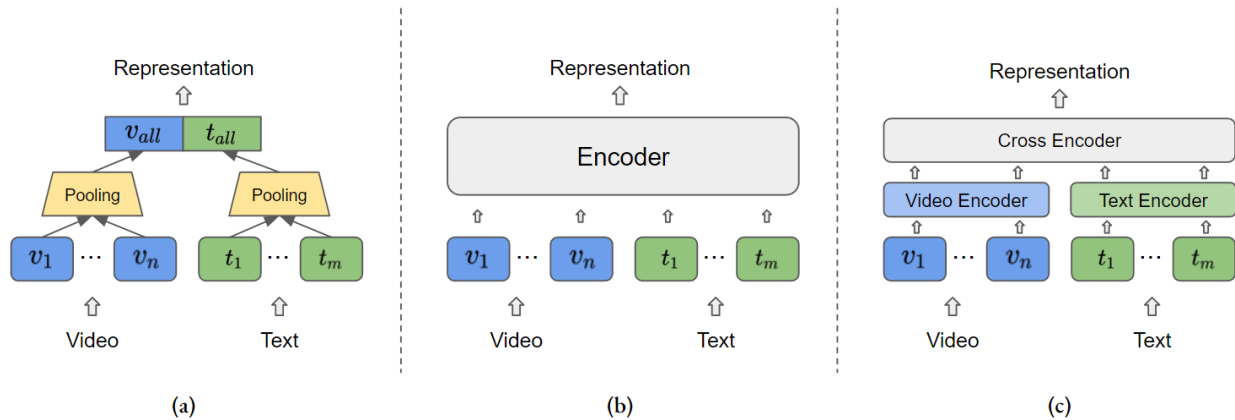


Figure 2.1: **Methods of Fusing Video and Text Features** – (a) Direct Concatenation; (b) Extending BERT to include video; (c) Cross-encoder.

HowTo100M[23], VATT[1], and VideoCLIP[40] project video and text features onto the same representation space, to ensure that video and text of similar semantic meanings are close to each other in representation. A direct concatenation of the video and text features is used to represent their fusion. However, this method suffers from information loss in video and text feature generation. Temporal information within video and text are forgone as each modality pools its features into one large feature to represent the modality. VideoBERT[31] instead extends the classical BERT[8] model to take in video features, so the temporal information in both video and text are preserved and utilized in one transformer, but limited processing is done within each modality. Latest methods such as UniVL[18], HERO[14], CLIPBERT[12] combine the best of the both worlds. They use a two-stage setup, where each modality is processed first using a encoder to extract most useful information, and then all features are fed into a cross-encoder that can reason relationship between the video and text features. In this work, we use both modality-specific encoders and cross-encoder to fully extract video and text information.

To learn meaningful fused representations of video and text, several losses are exploited, including contrastive and non-contrastive losses. Contrastive losses help models learn pulling video and text of the similar semantic content closer in the joint representation space, and push dissimilar video and text further away. They rely on contrasting positive video-text pairs against negative video-text pairs. HowTo100M[23] uses max-margin ranking loss to maximize the distance of a mismatched pair of video and text, but because the randomly constructed negative samples can be noisy, contrasting one positive example with only one

negative example renders the ranking loss unstable. VATT[1] and VideoCLIP[40] use InfoNCE[25] and MIL-NCE losses[21]. Given one or multiple instances of positive example, i.e. matching video and text, and multiple examples of negative examples, i.e. mismatching video and text, NCE uses cross-entropy loss to select the positive pairs within all examples. By using multiple negative examples, the variance of noise in gradient updates is reduced. On the other hand, VideoBERT[31] proposes to use a non-contrastive loss to learn joint video-and-text distribution. It uses Masked Language Modeling(MLM)[8] to model the distribution of each individual video/text tokens given the rest of the tokens, which compose the semantic context. Recent works have found it beneficial to combine contrastive losses with MLM losses of token prediction using context. UniVL[18] combines video-text alignment NCE loss, MLM with context and masked frame modeling with context. HERO[14] combines local alignment NCE loss of video and subtitle with masked language and frame modeling. In this work, we use both contrastive and masked prediction losses. Our contrastive loss is a variant of NCE loss with regularization to reduce the amount of noise in negative sampling, and masked prediction loss is a variant of MLM given context.

2.2 Misinformation Detection

Many benchmarks and evaluation datasets have been established for detecting misinformation, targeting different areas. BS Detector, LIAR, and FakeNewsNet manually label news articles for fake news detection. BuzzFeedNews, CREDBANK, and TW_info[11] are labeled dataset of misinformation in social media posts. However, very few video-and-text misinformation datasets are publicly available for evaluation. At the time of writing this report, Fake Video Corpus[27] is the only video-text misinformation dataset that is publicly available. It focuses on fake video detection in the context of text, and only contains 200 fake videos and 180 real videos. In this work, we publish a manually labeled video-text misinformation dataset with 401 tweets, of 317 matching tweets and 84 mismatching tweets.

There is a large body of literature on detecting multi-modal semantic inconsistencies. Luo et al. [17] leverage the expressiveness of a large pre-trained contrastive model CLIP [28] to classify misinformation based on retrieval. While methods based on billion parameter scale models can be powerful, many users do not have access to the compute or data required to train such models. Recently, several methods have been proposed for detecting differences in image and text semantics. Singhal et al. [30] leverages a learned joint embedding space. However, it requires both labeled positives and negatives in the data, and is specifically restricted to the news domain. Pan et al. [26] and Mayank et al. [19] focus explicitly on the textual description, detecting fake news using knowledge-graph based approaches. Tan et al. [33] and Fung et al. [9] focus on detecting synthetically generated news using text, image and knowledge element extraction. While these methods are feasible in situations where large labeled datasets of paired and unpaired semantic images and text exist, they do not transfer well to the more complex and sparsely labeled video domain.

In the video/text domain, Shang et al. [29] use video, audio, text and metadata in

TikTok videos to detect misleading COVID-19 video posts by fusing features from pre-trained models. However, they do not leverage the power of representation learning, and their method requires strong supervision, leading to generalization issues in low-resource domains. McCrae et al. [20] extract video, text, and named entity information from a news post, and utilize pretext-task learning on randomly permuted data to supervise a LSTM-based model. Since the method directly fuses video and text at each key-frame through concatenation, and does not learn a joint model of video and text, the model is unable to build complex joint representations. In this work, we use state-of-the-art video-language understanding methods, including cross-encoder, NCE loss and MLM, and achieve 9.03% higher accuracy on random mismatch detection, 14.96% higher accuracy on labeled misinformation detection as compared to McCrae et al. [20]. We further discover that without joint representation, state-of-the-art methods are merely detecting topic mismatch in video and text, as shown in Section 6.5. On detecting random mismatch on tweets of the same topic, our methods greatly outperform state-of-the-art methods, and exhibit capabilities in detecting semantic mismatch, rather than only topic mismatch.

Chapter 3

Video-text Misinformation Detection Pipeline

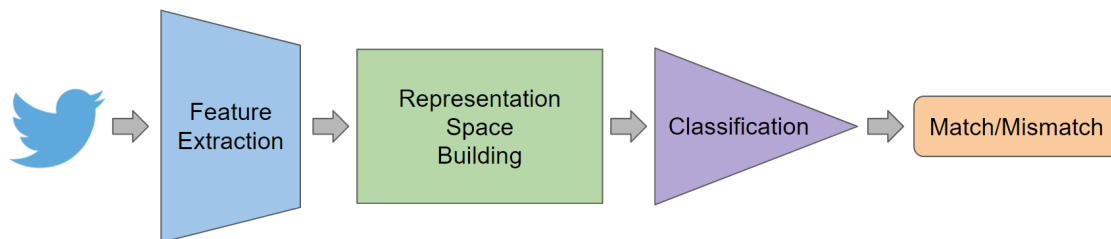


Figure 3.1: **Misinformation Detection Pipeline**

Our proposed overall pipeline is shown in Figure 3.1. Given a video post consisting of a video and a corresponding text description, we first use pre-trained models to extract video and text features. Then, the features are mapped to a common representation space so that video and text can be reasoned together. Lastly, we use the mapped video and text representations to classify whether the pair of video and text is a match or a mismatch.

We convert all videos into 10-fps, and break each video into segments of 32 frames. We use S3D [22], pre-trained on activity recognition, to extract one 512-dimensional video feature per video segment, resulting in $v = (v_1, \dots, v_n)$ $v_i \in \mathbb{R}^{512}$. For the text input, we use DeBERTa-v3-Large [10], pre-trained using MLM, to extract token-level features, where a text feature is generated corresponding to each text token. This results in $t = (t_1, \dots, t_m)$ $t_i \in \mathbb{R}^{1024}$.

While it is possible to learn and fine-tune the feature extraction methods on our dataset, this approach is too costly in terms of compute resources. In this work, we assume the feature extraction methods extract sufficiently useful information from both video and text. This allows us to focus more on building a better representation space and obtain better classification results without being concerned about the quality of the feature extractors. With that said, we do explore the effect of feature extractors on our classification results, and show in Section 6.4.1 that "better" feature extractor does lead to better performance.

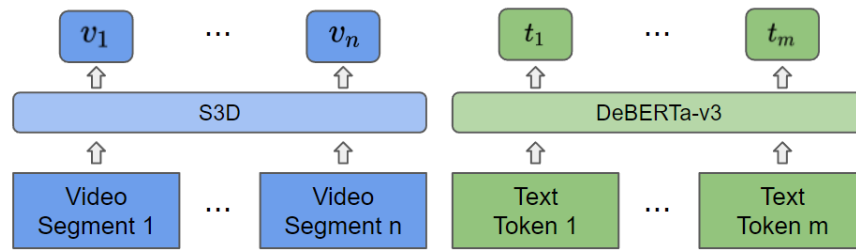


Figure 3.2: **Feature Extraction Using S3D and DeBERTa-v3**

Given a sequence of extracted video features and a sequence of extracted text features, we project these features onto a learned representation space. We discuss our methods of learning such a video and text representation space in Chapters 4 and 5. In Sections 4.5 and 5.3, we show that learning a representation space is helpful towards downstream classification.

Due to the lack of large labeled misinformation dataset in the video and text domain, we use self-supervised methods to train our models. We use the auxiliary task of random mismatch to train our model to predict whether the content of a video and a text description are matched or mismatched. Given a video, a matched text is the text accompanying the video post; we create a mismatched pair by randomly selecting another text from the dataset. Our labels are balanced, since we create a mismatched pair with 50% probability. A positive correlation exists between the performance of random mismatch and performance of misinformation detection in labeled dataset, shown later in Tables 6.1 and 6.2. Therefore, the task of detecting random mismatch transfers well to misinformation detection.

Chapter 4

Contrastive Learning for Misinformation Detection

In this chapter, we will explore using contrastive learning to build a video and text representation space. The main idea behind contrastive learning is to learn to project features onto a representation space, such that elements of matching semantic meanings are close to each other, while those of mismatching semantic meanings are away from each other.

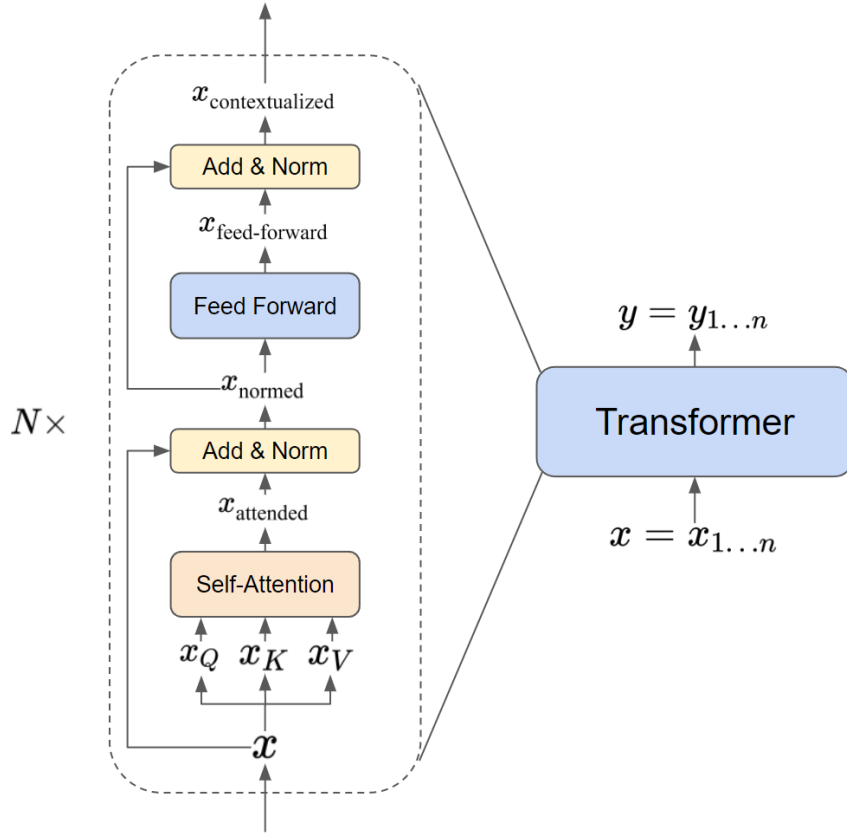
4.1 Background on Transformer

Transformer encoders[36] are often used in video and language processing to extract the temporal information. Figure 4.1 shows the transformer layer architecture. Given a sequence of n input features, each of dimension d_k , $x = x_{1\dots n} \in \mathbb{R}^{d_k}$, a transformer encoder applies self-attention to the features, and outputs embedded features $y = y_{1\dots n} \in \mathbb{R}^{d_k}$. A transformer encoder is a stack of N transformer layers. We choose $d_k = 1024$, $N = 2$ for all transformers used in this work. Each transformer layer contains self-attention, residual connection, layer normalization[2], and feed-forward, as shown in Figure 4.1.

Given input features $x = x_{1\dots n}$, self-attention linearly projects each feature into three vectors, and uses softmax attention to compute a weighted sum of x_V based on projected x_Q and x_K :

$$\begin{aligned}
 x_Q &= xW_Q \\
 x_K &= xW_K \\
 x_V &= xW_V \\
 x_{\text{attended}} &= \text{softmax}\left(\frac{x_Q x_K^T}{\sqrt{d_k}}\right)x_V
 \end{aligned} \tag{4.1}$$

where $W_Q, W_K, W_V \in \mathbb{R}^{1024 \times 1024}$ are learned parameter matrices. After self-attention,

Figure 4.1: **Transformer Layer Architecture**

residual connection is added to x_{attended} , and layer normalization is performed:

$$x_{\text{normed}} = \text{LayerNorm}(x + x_{\text{attended}}) \quad (4.2)$$

Next, we apply a 2-layer feed-forward network with ReLU activation to x_{normed} , and further apply residual connection and layer normalization:

$$\begin{aligned} x_{\text{feed-forward}} &= \max(0, x_{\text{normed}}W_1 + b_1)W_2 + b_2 \\ x_{\text{contextualized}} &= \text{LayerNorm}(x_{\text{normed}} + x_{\text{feed-forward}}) \end{aligned} \quad (4.3)$$

where $W_1, W_2 \in \mathbb{R}^{1024 \times 1024}$ are learned weight matrices, and $b_1, b_2 \in \mathbb{R}^{1024}$ are learned bias vectors. This x to $x_{\text{contextualized}}$ process is a transformer layer. Transformer encoder obtains y by having x pass through a stack of N transformer layers.

4.2 Feature Aggregation

Since video and text are both sequences of data, it is important to process the temporal information within each modality, rather than naïvely averaging all features together. We

use transformers[36] to embed features of each modality and to retrieve temporal information, and a global mean pooling to aggregate the information.

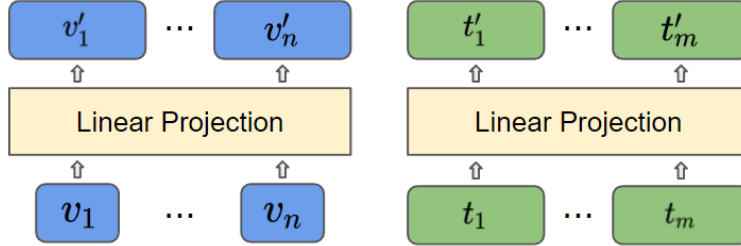


Figure 4.2: **Project Video and Text Features to the Same Dimension**

Given video features of $v = (v^1, \dots, v^n)$, $v^i \in \mathbb{R}^{512}$ and $t = (t^1, \dots, t^m)$, $t^i \in \mathbb{R}^{1024}$, we first use a linear projection to project all features onto the same dimension space \mathbb{R}^{1024} , as shown in Figure 4.2.

$$\begin{aligned} v' &= W_v v \\ t' &= W_t t \end{aligned} \quad (4.4)$$

where $W_v \in \mathbb{R}^{1024 \times 512}$ and $W_t \in \mathbb{R}^{1024 \times 1024}$ are learned parameters. Next, as seen in Figure 4.3, video features $v' = (v'_1, \dots, v'_n)$ and text features $t' = (t'_1, \dots, t'_m)$ each go through a transformer to create features of video and text context, $h = (h_1, \dots, h_n)$, $h_i \in \mathbb{R}^{1024}$ and $k = (k_1, \dots, k_m)$, $k_i \in \mathbb{R}^{1024}$, respectively. We next apply global mean pooling on features of each modality, h and k , and retrieve aggregated modality features, v^{embed} and t^{embed} :

$$\begin{aligned} v^{\text{embed}} &= \frac{1}{n} \sum_{i=1}^n h_i \\ t^{\text{embed}} &= \frac{1}{m} \sum_{i=1}^m k_i \end{aligned} \quad (4.5)$$

4.3 Contrastive Learning

For the architecture shown in Figure 4.3, we discuss and apply two different methods of learning the joint video and text representation using Contrastive Learning: (a) Contrastive loss[6] and (b) Noise Contrastive Estimation loss[25]. We discuss each in the following sections.

4.3.1 Contrastive Loss

Our first method uses Contrastive loss [6] to build the representation space of video and text. Given the embedded video and text features $v^{\text{embed}} \in \mathbb{R}^{1024}$ and $t^{\text{embed}} \in \mathbb{R}^{1024}$, we apply a

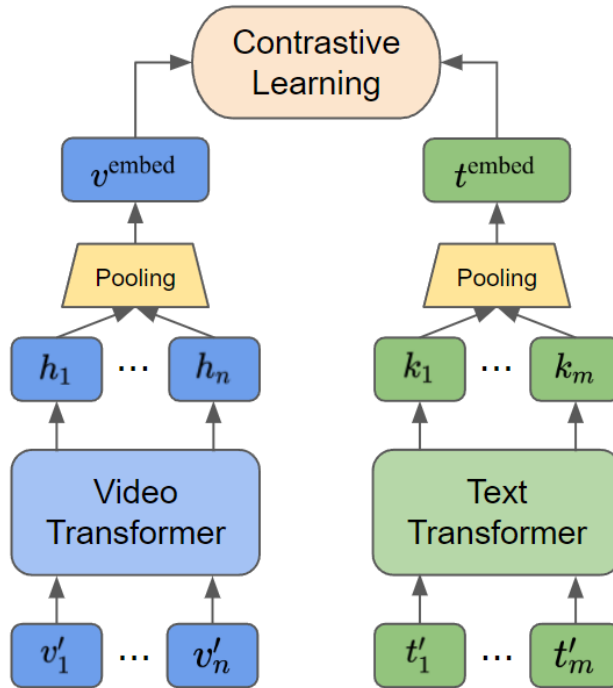


Figure 4.3: **Contrastive Learning** – separate transformers for each modality, aggregation through mean pooling, contrastive learning on aggregated features.

cosine embedding loss, L_{\cos} to construct the representation space of video and text:

$$\mathcal{L}_{\cos}(v^{\text{embed}}, t^{\text{embed}}, y) = \begin{cases} 1 - \cos(v^{\text{embed}}, t^{\text{embed}}) & y=0 \\ \max(0, \cos(v^{\text{embed}}, t^{\text{embed}})) & y=1 \end{cases} \quad (4.6)$$

where y denotes the label of the pair of video and text, 0 for a match, and 1 for a mismatch. Cosine embedding loss L_{\cos} encourages the vector angle between a matching pair of video and text to be smaller, and the angle of a mismatching pair to be larger. This allows the models to learn v^{embed} and t^{embed} .

4.3.2 Noise Contrastive Estimation Loss

One problem with Contrastive Loss is that it only considers one instance of positive or negative sample at a time. The process of constructing negative examples through random mismatch makes the authenticity of the negative sample noisy. For example, it is likely to generate one negative sample that is in fact matching, thereby learning on this sample would push the originally close video and text away from each other. Therefore, to reduce noise presented in the negative samples, we sample and learn on multiple negative samples at a time using a variant of Noise Contrastive Estimation(NCE) loss[25], similar to NCE loss used in CLIP[28].

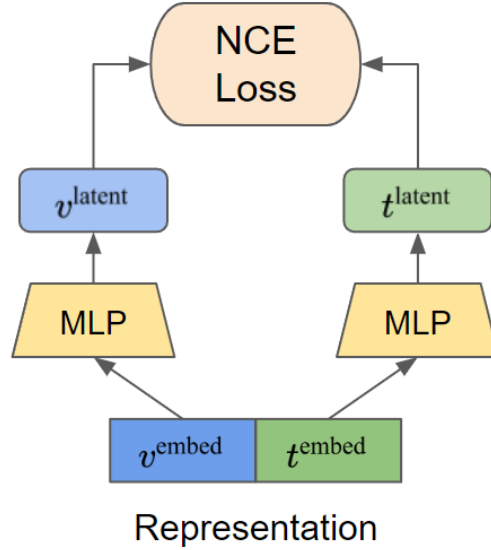


Figure 4.4: **Projection of Video and Text Representations for NCE Loss.**

We first project each representation $v^{\text{embed}}, t^{\text{embed}}$ into a latent space $v^{\text{latent}}, t^{\text{latent}} \in \mathbb{R}^{1024}$, as illustrated in Figure 4.4. It has been shown in recent self-supervision studies[4, 5] that this approach learns a more disentangled representation space. We use 2-layer MLPs for projecting $v^{\text{embed}}, t^{\text{embed}}$ to a latent space, but still use the before-projection features as our representation. After projection, we obtain:

$$\begin{aligned} v^{\text{latent}} &= W_{2,v} \max(0, W_{1,v} v^{\text{embed}}) \\ t^{\text{latent}} &= W_{2,t} \max(0, W_{1,t} t^{\text{embed}}) \end{aligned} \quad (4.7)$$

where $W_{1,v}, W_{1,t} \in \mathbb{R}^{1024 \times 1024}$ are learned weight matrices in first layers, $W_{2,v}, W_{2,t} \in \mathbb{R}^{1024 \times 1024}$ are learned weight matrices in second layers, and $v^{\text{latent}}, t^{\text{latent}} \in \mathbb{R}^{1024}$ are inputs to the NCE loss.

Given a batch of B matching video and text, we have embedded video and text features $v_i^{\text{latent}}, t_i^{\text{latent}} \in \mathbb{R}^{1024}; i \in [1, B]$. We contrast each matching pair with all other mismatching pairs for both video and text. Specifically, given v_i^{latent} , the matching text feature is t_i^{latent} , and all other mismatching text features are $t_j^{\text{latent}}, j \neq i, j \in [1, B]$. Out of all B text features, we learn to classify the text feature matching v_i^{latent} . Therefore, we minimize the cross entropy of each video feature and its matching text feature, versus other text features in the batch. Following conventions used in CLIP[28], we l_2 -normalize $v^{\text{latent}}, t^{\text{latent}}$ first:

$$\begin{aligned}\hat{v}^{\text{latent}} &= \frac{v^{\text{latent}}}{\|v^{\text{latent}}\|_2} \\ \hat{t}^{\text{latent}} &= \frac{t^{\text{latent}}}{\|t^{\text{latent}}\|_2}\end{aligned}\tag{4.8}$$

and also scale their dot product using a learned temperature parameter T . This results in the following video-to-text loss function:

$$\mathcal{L}_{\text{video} \rightarrow \text{text}} = -\frac{1}{B} \sum_{i=1}^B \left(\log \frac{\exp(T \hat{v}_i^{\text{latent}} \cdot \hat{t}_i^{\text{latent}})}{\sum_{j=1; i \neq j}^B \exp(T \hat{v}_i^{\text{latent}} \cdot \hat{t}_j^{\text{latent}})} \right)\tag{4.9}$$

We also learn the reverse loss, namely given t_i^{latent} , we learn to classify which video feature is matching it among all B video features:

$$\mathcal{L}_{\text{text} \rightarrow \text{video}} = -\frac{1}{B} \sum_{i=1}^B \left(\log \frac{\exp(T \hat{t}_i^{\text{latent}} \cdot \hat{v}_i^{\text{latent}})}{\sum_{j=1; i \neq j}^B \exp(T \hat{t}_i^{\text{latent}} \cdot \hat{v}_j^{\text{latent}})} \right)\tag{4.10}$$

We optimize the mean of these two losses to obtain:

$$\mathcal{L}_{NCE} = \frac{1}{2} \mathcal{L}_{\text{text} \rightarrow \text{video}} + \frac{1}{2} \mathcal{L}_{\text{video} \rightarrow \text{text}}\tag{4.11}$$

4.4 Video-text Fusion Methods

To detect misinformation, we explore two different methods of fusing video and text representations. In the next two subsections, we describe each method, (a) direct concatenation(DC) and (b) cross-transformer(CT), as illustrated in Figure 4.5.

4.4.1 Direct Concatenation(DC)

The architecture for direct concatenation approach is shown in Figure 4.5(a). We concatenate v^{embed} and t^{embed} to obtain the joint representation $R_{\text{concat}} \in \mathbb{R}^{2048}$:

$$R_{\text{concat}} = v^{\text{embed}} \oplus t^{\text{embed}}\tag{4.12}$$

and use a 4-layer MLP over the joint representation R_{concat} to regress the probability of matching \hat{y} :

$$\hat{y} = \sigma(\text{MLP}(R_{\text{concat}}))\tag{4.13}$$

which we supervise with binary cross-entropy loss L_{BCE} :

$$\mathcal{L}_{\text{BCE}} = y \cdot \log(\hat{y}) + (1 - y) \cdot \log(1 - \hat{y})\tag{4.14}$$

We refer to this architecture as DC in evaluations.

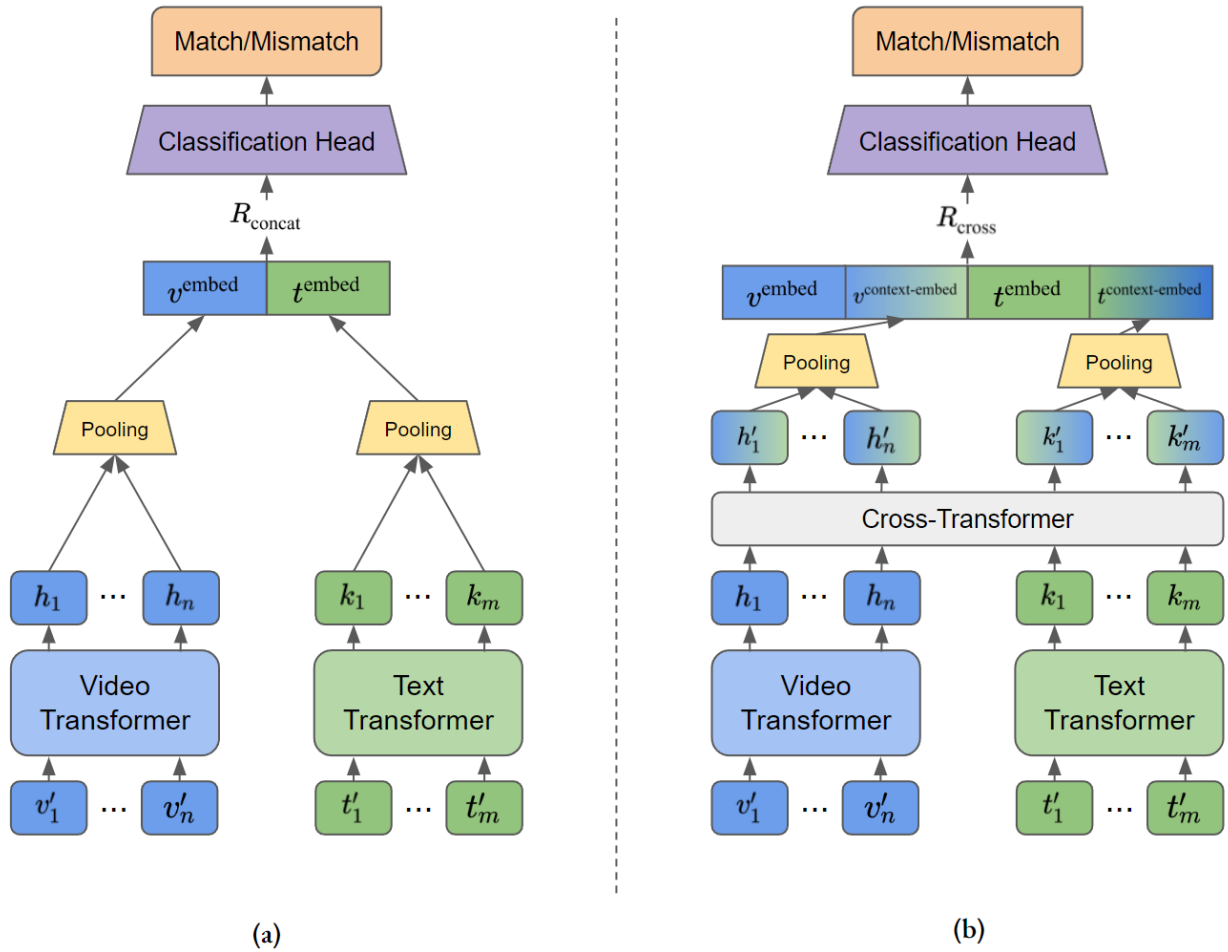


Figure 4.5: **Fusion Methods in Using Learned Representation Space** – (a) Direct Concatenation(DC); (b) Cross-Transformer(CT).

4.4.2 Cross-Transformer(CT)

The architecture for cross-transformer is shown in Figure 4.5(b). To fully capture the temporal information in each modality, and allow video and text features to interact with each other, we use one cross-transformer that takes in the last hidden states of both video and text transformers, as shown in Figure 4.5(b). The transformer takes in h_1, \dots, h_n and k_1, \dots, k_m , the hidden states before feature aggregation in Section 4.2, and uses transformer attention mechanism to contextualize them to create new hidden states of video and text, h'_1, \dots, h'_n and k'_1, \dots, k'_m , respectively. We then apply global mean pooling on features of each modality

Loss	Accuracy	Precision	Recall	M-Precision	M-Recall
CT - \mathcal{L}_{BCE}	80.85%	78.92%	84.28%	83.08%	77.42%
CT - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{cos}}$	81.45%	79.53%	84.76%	83.63%	78.13%
CT - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$	85.43%	85.24%	85.44%	85.62%	85.43%

Table 4.1: **Effect of Contrastive Learning on Random Mismatch Detection** – NCE loss improves accuracy by 4.58%, compared to only using BCE loss to learn classification.

to obtain contextualized aggregated modality features, $v^{\text{context-embed}}$ and $t^{\text{context-embed}}$:

$$\begin{aligned}
 v^{\text{context-embed}} &= \frac{1}{n} \sum_{i=1}^n h'_i \\
 t^{\text{context-embed}} &= \frac{1}{m} \sum_{i=1}^m k'_i
 \end{aligned}
 \tag{4.15}$$

We concatenate both contextualized and raw feature embeddings to obtain fused representation $R_{\text{cross}} \in \mathbb{R}^{4096}$:

$$R_{\text{cross}} = v^{\text{embed}} \oplus v^{\text{context-embed}} \oplus t^{\text{embed}} \oplus t^{\text{context-embed}}
 \tag{4.16}$$

With representation R_{cross} , we use the same MLP and \mathcal{L}_{BCE} setup as in Equation (4.14) to further classify misinformation. We refer to this architecture as CT in evaluations.

4.5 Evaluations

We train four different models on Twitter 1M dataset to evaluate contrastive learning’s effectiveness on misinformation detection. The four models are (1) CT trained with \mathcal{L}_{BCE} denoted by CT - \mathcal{L}_{BCE} , (2) CT trained with $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{cos}}$ denoted by CT - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{cos}}$, (3) CT trained with $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$ denoted by CT - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$, and (4) DC trained with $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$ denoted by DC - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$.

4.5.1 Contrastive Learning

We compare CT model with three different losses in Tables 4.1 and 4.2. Precision and Recall are measured in terms of matching-post detection, where a matching post is considered as the positive class. M-Precision and M-Recall are precision and recall measured in terms of mismatching-post, or misinformation detection, where a mismatching post is considered as the positive class. We find it helpful to consider both when evaluating our methods.

Loss	Accuracy	Precision	Recall	M-Precision	M-Recall
CT - \mathcal{L}_{BCE}	74.06%	80.17%	89.27%	29.17%	16.67%
CT - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{cos}}$	74.06%	81.05%	87.70%	32.76%	22.62%
CT - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$	74.81%	81.03%	88.96%	33.96%	21.43%

Table 4.2: **Effect of Contrastive Learning on Labeled Misinformation Detection** – NCE loss improves accuracy by 0.75%, compared to only using BCE loss to learn classification.

Fusion Method	Accuracy	Precision	Recall	M-Precision	M-Recall
DC - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$	84.36%	85.45%	83.00%	83.31%	85.73%
CT - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$	85.43%	85.24%	85.44%	85.62%	85.43%

Table 4.3: **Effect of Fusion Method on Random Mismatch Detection** – Cross-Encoder performs 1.07% higher than direct concatenation.

CT - \mathcal{L}_{BCE} does not construct any representation space, and simply learns to classify misinformation. If we add \mathcal{L}_{cos} to learn a noisy representation space, the model performance improves slightly by 0.60% in accuracy of random mismatch detection, and stays the same during labeled misinformation detection. However, by adding \mathcal{L}_{NCE} to learn a robust representation space, the misinformation detection accuracy improves by 4.58% and 0.75%, respectively. Noticeably, both M-Precision and M-Recall improve over 4% in labeled misinformation detection, indicating better efficiency and accuracy at filtering social media posts for misinformation.

4.5.2 Comparison of Fusion Methods

We compare DC - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$ and CT - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$ downstream detection accuracies. In Tables 4.3 and 4.4, we observe that cross-encoder outperforms direct concatenation in accuracy by 1.07% on random mismatch detection and 1.24% on labeled misinformation detection. This is due to its use of temporal information in combining video and text features. We choose to use cross-encoder as the default setup in future ablation studies in Chapter 6.

Fusion Method	Accuracy	Precision	Recall	M-Precision	M-Recall
DC - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$	73.57%	81.49%	86.12%	33.33%	26.19%
CT - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$	74.81%	81.03%	88.96%	33.96%	21.43%

Table 4.4: **Effect of Fusion Method on Labeled Misinformation Detection** – Cross-Encoder performs 1.24% higher than direct concatenation.

Chapter 5

Masked Language Modeling for Misinformation Detection

In this chapter, we explore learning video and text representation using MLM proposed in BERT [8]. The idea behind MLM is to mask a small portion of input tokens and learn to reconstruct them using the rest of the input. For example, given the sentence "This [MASK] a radio." with the word "is" being masked, the model learns to predict what the masked word is. In our implementation, we mask 30% of the text tokens and reconstruct them using the information in both video tokens and the rest of the text tokens. We follow BERT[8]'s convention in masking, where we replace the masked token with [MASK] 80% of the time, with another random token 10% of the time, and itself 10% of the time.

5.1 Masked Language Modeling

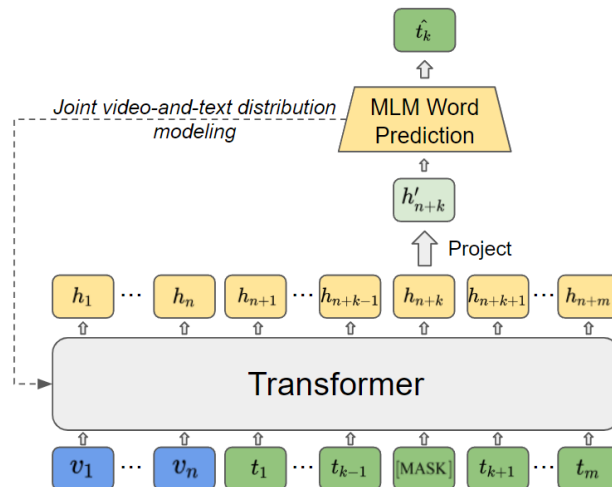


Figure 5.1: Masked Language Modeling – Model Architecture

Our proposed model architecture is shown in Figure 5.1. We train one single transformer to approximate the maximum log-likelihood of each text token given its text context and the video,

$$E = \sum_i^m \log(\mathbb{P}(t_i | t_{j \neq i}, v_{1..n}; \theta)) \quad (5.1)$$

where E is the total log-likelihood to be maximized, $t_{1..m}$ are all m text tokens in video description, $v_{1..n}$ are video tokens, and θ represents parameters of the transformer, which are optimized through the MLM objective from Devlin et al. [8]. Video and text features, $t_{1..m}, v_{1..n}$, are extracted using S3D and DeBERTa-v3, as shown in Figure 3.2.

To model the data, we use WordPiece [39] to tokenize each word of our text description with vocabulary size of 30,522, and embed using a learned text embedding to obtain token embeddings $t_{1..m} \in \mathbb{R}^{768}$. We project our video features $v_{1..n}$ onto the same dimension \mathbb{R}^{768} using a 2-layer MLP. Then, we randomly replace 30% of the text tokens with a special token [MASK]. We construct our entire input embedding sequence as:

$$\begin{aligned} \text{video} &= (v_1, \dots, v_n) \\ \text{masked-text} &= (t_1, \dots, t_{k-1}, [\text{MASK}], t_{k+1}, \dots, t_m) \\ \text{input} &= \text{video} \oplus \text{masked-text} \\ \text{position} &= \text{Positional-Embedding}(\text{input}) \\ \text{ordered-input} &= \text{position} + \text{input} \end{aligned} \quad (5.2)$$

We first add learned positional embeddings [36] to our input embedding sequence to capture the temporal order in video and text. Positional embeddings are gathered through indexing directly in a learned matrix $P \in \mathbb{R}^{260 \times 768}$, where 260 is the maximum input sequence length. For instance, positional embedding of position 0 is the first row of this matrix, a vector of \mathbb{R}^{768} . We sum the original input and the positional embeddings to obtain an ordered input.

Next, we apply a transformer encoder with hidden dimension 768, feed-forward dimension 1024, and 2 layers on the ordered input to generate hidden states $h_{1..(n+m+1)} \in \mathbb{R}^{768}$. As shown in Figure 5.1, each masked token's corresponding hidden state, h_{n+k} , is projected to the dimension of vocabulary size, $h'_{n+k} \in \mathbb{R}^{30,522}$. During training, we ask our model to reconstruct the original text tokens that were replaced by [MASK] tokens to learn each word's distribution within the context of the social media video post, $\mathbb{P}(t_i | \text{masked-text}, v_{1..n})$. We use the cross-entropy reconstruction loss as our masked language modeling loss:

$$\mathcal{L}_{\text{MLM}} = -\frac{1}{|M|} \sum_{k \in M} \log \frac{\exp(h'_{n+k, \hat{t}_k})}{\sum_{t \in \text{vocab}; t \neq \hat{t}_k} \exp(h'_{n+k, t})} \quad (5.3)$$

where M is the set of indices of text tokens masked in the input, \hat{t}_k is the original text token of the masked k th text token, and $h'_{n+k,t}$ is the logit of the token t , taken from the projected hidden state h'_{n+k} of the masked k th text token.

5.2 Learned Representation Space

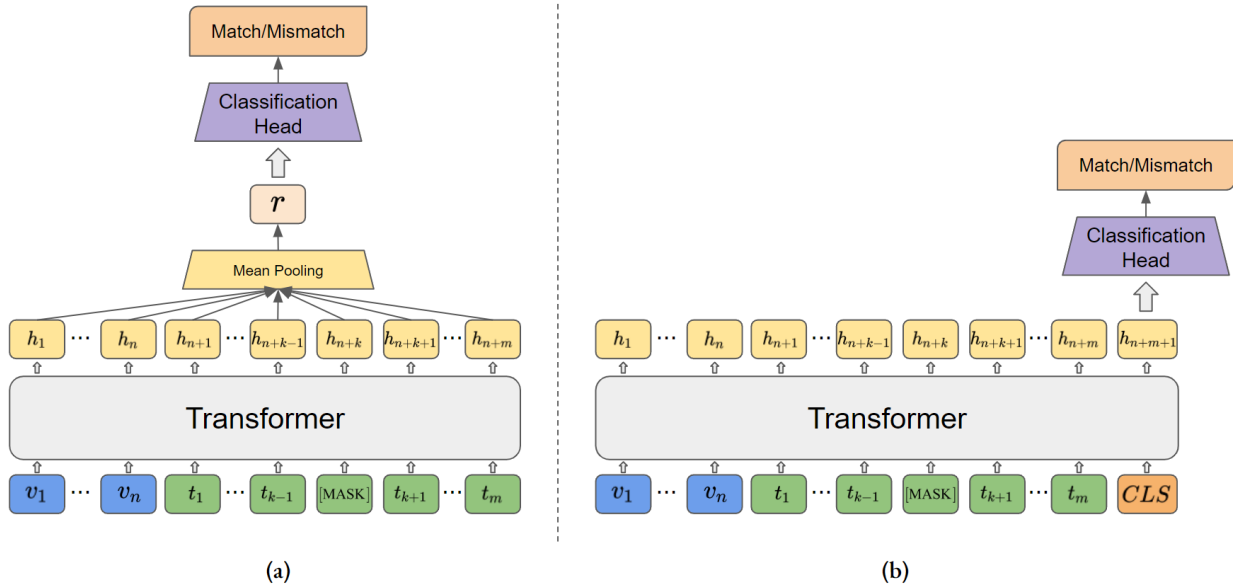


Figure 5.2: **Methods of Extracting Representation from Learned Representation space** – (a) Global Mean Pooling; (b) Attention.

To detect misinformation, we experiment with two different methods to extract learned representations: (a) Global Mean Pooling and (b) Attention, as illustrated in Figure 5.2.

5.2.1 Global Mean Pooling(GMP)

Learned representation through global mean pooling is shown in Figure 5.2(a). We apply global mean pooling to hidden states of the transformer $h_{1\dots(n+m)} \in \mathbb{R}^{768}$, to compute one aggregated representation $r \in \mathbb{R}^{768}$:

$$r = \frac{1}{n+m} \sum_{i=1}^{n+m} h_i \quad (5.4)$$

We then apply a 4-layer MLP on r and learn classification using binary cross-entropy loss, \mathcal{L}_{BCE} , shown in Equation (4.14). We refer to this architecture as GMP in evaluations.

Loss	Accuracy	Precision	Recall	M-Precision	M-Recall
CLS - \mathcal{L}_{BCE}	80.12%	83.23%	75.46%	77.55%	84.79%
CLS - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{MLM}}$	81.53%	83.33%	78.82%	79.91%	84.23%

Table 5.1: **Effect of Masked Language Modeling on Random Mismatch Detection** – MLM loss improves accuracy by 1.41%, compared to only using BCE loss to learn classification.

5.2.2 [CLS] Extraction through Attention(CLS)

Representation extraction architecture through attention is shown in Figure 5.2(b). In order to utilize transformer’s attention mechanism in Equation (4.1), we further append a learned input token, $[\text{CLS}] \in \mathbb{R}^{768}$, at the end of input sequence to extract all video-text information through encoding. The input to the transformer in Figure 5.2(b) now becomes:

$$\text{input} = \text{video} \oplus \text{masked-text} \oplus [\text{CLS}] \quad (5.5)$$

The $[\text{CLS}]$ token is able to pay attention to both video and text to extract information needed for mismatch classification. The last hidden state of transformer output, h_{n+m+1} , is the corresponding output of the $[\text{CLS}]$ token. We further apply a 4-layer MLP on h_{n+m+1} and compute binary cross-entropy loss using Equation (4.14). We refer to this architecture as CLS in evaluations.

5.3 Evaluations

We train an additional three models on Twitter 1M dataset to investigate effectiveness of MLM and representation extraction methods towards misinformation detection. The three additional models are: (a) CLS extraction through attention trained with \mathcal{L}_{BCE} , denoted by CLS - \mathcal{L}_{BCE} ; (b) CLS extraction through attention trained with $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{MLM}}$, denoted by CLS - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{MLM}}$; and (c) GMP trained with $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{MLM}}$, denoted by GMP - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{MLM}}$.

5.3.1 MLM Loss

We compare models CLS - \mathcal{L}_{BCE} and CLS - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{MLM}}$ in Tables 5.1 and 5.2 for random and labeled mismatch respectively. As seen, CLS with MLM loss does improve model’s random mismatch and labeled misinformation detection accuracy by 1.41% and 3.99% respectively. This shows that learning text distribution given video is helpful in detecting misinformation.

Loss	Accuracy	Precision	Recall	M-Precision	M-Recall
CLS - \mathcal{L}_{BCE}	68.33%	83.22%	75.08%	31.30%	42.86%
CLS - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{MLM}}$	72.32%	82.39%	82.65%	33.73%	33.33%

Table 5.2: **Effect of Masked Language Modeling on Labeled Misinformation Detection** – MLM loss improves accuracy by 3.99%, compared to only using BCE loss to learn classification.

Extraction Method	Accuracy	Precision	Recall	M-Precision	M-Recall
GMP - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{MLM}}$	80.63%	83.91%	75.79%	77.93%	85.47%
CLS - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{MLM}}$	81.53%	83.33%	78.82%	79.91%	84.23%

Table 5.3: **Effect of Representation Extraction Method on Random Mismatch Detection** – [CLS] token outperforms global mean pooling by 0.9% in accuracy.

Extraction Method	Accuracy	Precision	Recall	M-Precision	M-Recall
GMP - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{MLM}}$	67.58%	81.91%	75.71%	28.70%	36.90%
CLS - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{MLM}}$	72.32%	82.39%	82.65%	33.73%	33.33%

Table 5.4: **Effect of Representation Extraction Method on Labeled Misinformation Detection** – [CLS] token outperforms global mean pooling by 4.74% in accuracy.

5.3.2 Comparison of Representation Extraction Methods

We also compare the representation extraction methods used for misinformation detection. We compare CLS - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{MLM}}$ with GMP - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{MLM}}$ in Tables 5.3 and 5.4. Results in Table 5.3 show that CLS is slightly better for random mismatch, with an improvement in accuracy of 0.9%. In Table 5.4, we see a much larger improvement for CLS in labeled misinformation detection accuracy, improving by 4.74%. We use CLS by default in future ablation studies in Chapter 6.

5.4 Unsupervised Misinformation Grounding

There are two deficiencies in our current misinformation detection models. (1) Our models would not be able to provide any explanations to the classifications they make, and (2) there is always a distribution shift between our random-mismatch labels and the actual

misinformation detection labels. To address these issues, we investigate an unsupervised model that provides token-level feedback of where misinformation exists. This unsupervised model computes the ratio p of the probability of a token with the video input and the probability of a token without the video input as follows:

$$p_i = \frac{P(t_i|v_{1...n}, t_{j \neq i, j \in [1...m]})}{P(t_i|t_{j \neq i, j \in [1...m]})} \quad (5.6)$$

Ratio p greater than or equal to 1 implies text token to be consistent with the video content. However, if p is smaller than 1, the text token is more likely to exist without the video content, implying the text token to arise from misinformation.

This model uses the same architecture and loss as Section 5.1. During training, we train with both video and text 70% of the time, and train with only text 30% of the time. We achieve this by replacing all video tokens $v_{1...n}$ with a learned special token v_{mask} of the same dimension. During inference time, we collect the text token probability with and without video separately. We first run the model with the video input. We mask one text token at a time, collecting the probability of that token through softmax over the logits of the token predictor. This results in the numerator of the p-ratio. We then run the model without video input, i.e. with the v_{mask} tokens, mask and collect each text token’s probability again, resulting in the denominator of the p-ratio. Each text token’s p-ratio can now be calculated through division.

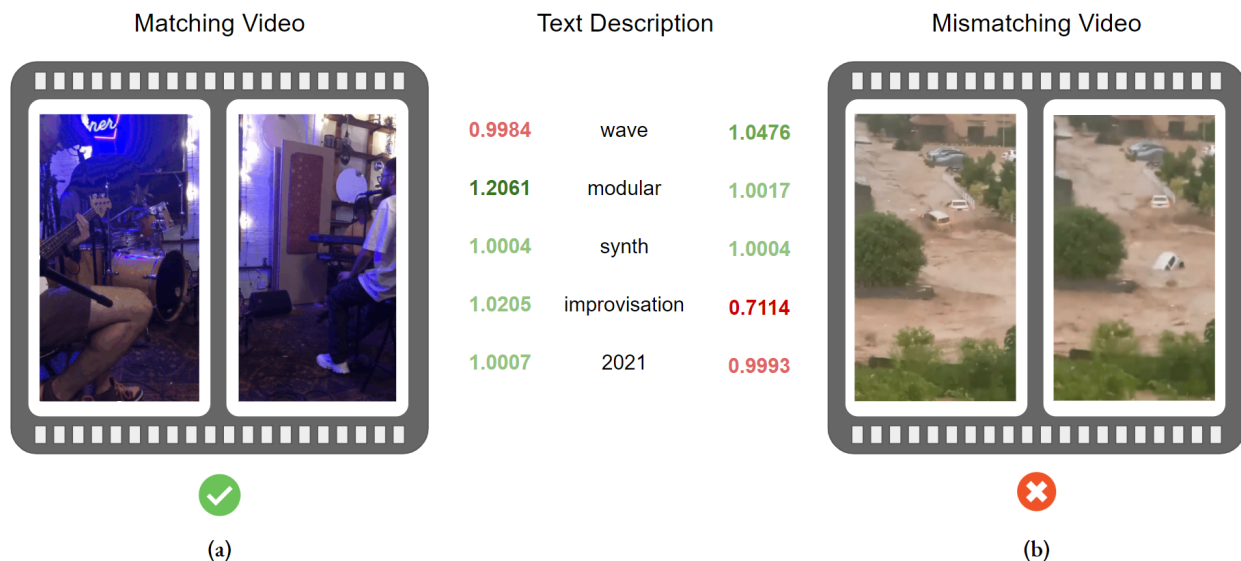


Figure 5.3: **Grounding Misinformation using Unsupervised Learning** – (a) Matching video; (b) Mismatching video. Each word is evaluated on whether it brings in misinformation to video as a description.

An example of the above scheme is shown in Figure 5.3. The Twitter post, "wave modular synth improvisation 2021", is shown in the middle. Its original matching video

Inference Method	Accuracy	Precision	Recall	M-Precision	M-Recall
Mean of p-ratios	52.00%	53.85%	28.00%	51.35%	76.00%
Majority vote of p-ratios	51.00%	57.14%	8.00%	50.54%	94.00%

Table 5.5: **Inference Methods of Unsupervised Learned Model on Random Mismatch Detection** – methods consistently predict more negative than positive, resulting in accuracy of a random guess, 50%.

is in Figure 5.3(a), showing a 4-person band in a jamming session. We speculate that "wave" refers to the bass, "modular" refers to the drumset, "synth" refers to the synthesizer, and "improvisation" refers to the rapper/singer. In Figure 5.3(b), we randomly select a mismatched video, depicting a flooding scene. The numbers in the middle of Figure 5.3 are computed p-ratios of each word, with/without the matching/mismatching video. In context of the matching video in Figure 5.3(a), the model struggles to relate the concept of "wave" in a jamming session, but shows that the other words are trust-worthy. Specifically, "modular" is much more convincing with the video. In context of the mismatching video in Figure 5.3(b), the model suggests that the word "wave" is matching with the video, which contains waves of flood. Importantly, it flags the word "improvisation" as misinformation with a low p-ratio, indicating that the word is less likely to be co-existing with a flooding video. In this way, if video is detected to be mismatched, the decision is grounded because the word "improvisation" implies misinformation.

We explore two different methods in making a misinformation decision given all p-ratios of text tokens. The first one takes a mean of all p-ratios, and the second one takes the majority vote of the p-ratios, with <1 values as negative and ≥ 1 values as positive. As shown in Table 5.5, we find that both methods result in accuracy of about 50%, where 94% the predictions made by majority vote method are negative when it comes to random mismatch detection. Both methods perform worse than random guess in labeled misinformation detection, as shown in Table 5.6.

One reason for such low performance is that during tokenization, a word is often broken into multiple pieces of tokens. During inference, we only mask out one token, which makes token prediction an easy task for the text-only model - the model is often over-confident about the token it predicts. This makes token probabilities with videos consistently smaller than token probabilities without videos, thus predicting more mismatches than it should. Future work in this direction could investigate better masking methods such as masking words/phrases rather than a single token.

Inference Method	Accuracy	Precision	Recall	M-Precision	M-Recall
Mean of p-ratios	35.91%	82.61%	23.97%	22.01%	80.95%
Majority vote of p-ratios	28.18%	74.58%	13.88%	20.18%	82.14%

Table 5.6: **Inference Methods of Unsupervised Learned Model on Labeled Misinformation Detection** – methods consistently predict more negative than positive, resulting in high Precision and M-Recall, but poor accuracy overall.

Chapter 6

Experiments and Ablation Studies

In this chapter, we discuss experiments and details, including dataset collection, performance comparison with previous works, and ablation studies on properties of our proposed misinformation detection pipeline.

6.1 Data Collection and Filtering

Distribution of Text and Video Lengths after Data Cleaning

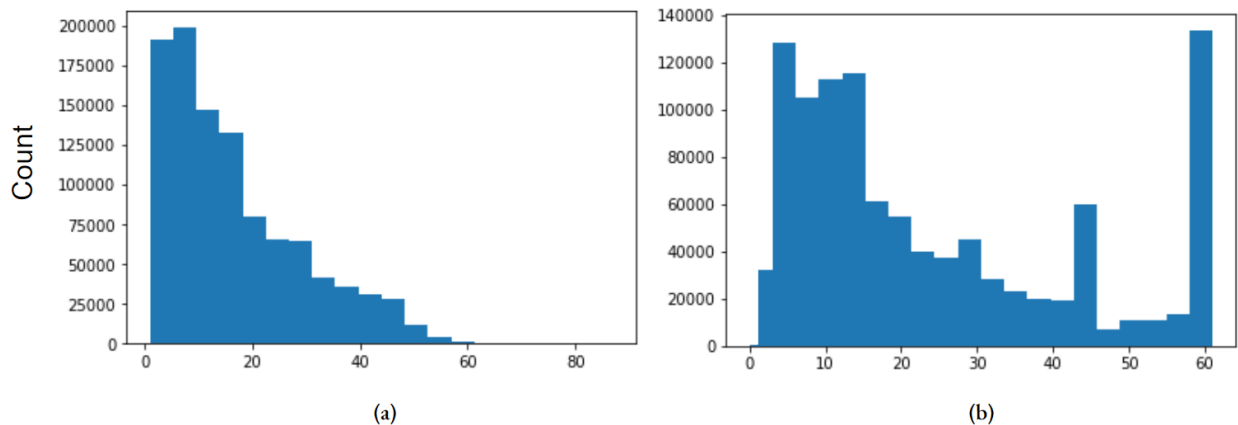


Figure 6.1: **Video and Text Length Distribution in Collected Twitter Dataset after Filtering.** – (a) Number of characters in Text; (b) Length of videos in seconds

We collect 1 million Twitter dataset using Twitter API. We only consider tweets that contain both video and text, and are not retweets/replies/quotes to other tweets. Our tweets' post time range from January 2021 to March 2022. To ensure an even data distribution, for each hour in post-time range, we collect 100 tweets that are posted within the hour.

For data cleaning, we remove any retweets/replies/quotes, and also tweets that are marked as "possibly_sensitive" and "possibly_sensitive_appeal", labeled by Twitter API. We then remove any tweets that contain a video shorter than 3 seconds or greater than 61 seconds, or a text shorter than 3 characters. Twitter also imposes a 280-character length upper limit. This removes 11% of the original collected tweets. After data cleaning, there remains 943,667 tweets in total, with an average video length of 25.04 seconds. The data distributions for video and text are shown in Figure 6.1. We use 80/10/10 split for train/validation/test.

Video Matcher

Please select if the video matches the description semantically.

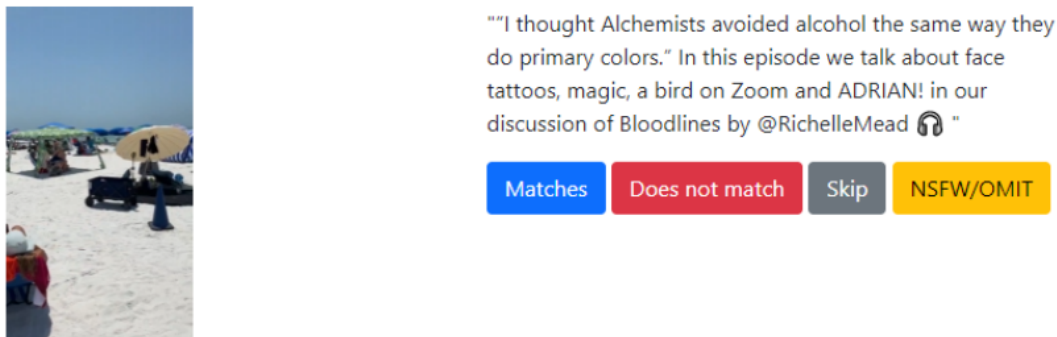


Figure 6.2: Misinformation Labeling System for Twitter Video Posts

We have also developed a social media misinformation labeling platform for labeled misinformation dataset collection. As shown in Figure 6.2, a worker labels a social media post as either matched or mismatched, or skips labeling the current post because of difficulty in understanding it. We use 3 workers, and each of them annotates a subset of the resulting 401 posts. This resulted in 317 matched video posts and 84 mismatched video posts. This is to be expected because most users post matched video and text on social media. Future work remains on improving this dataset and increasing its size.

6.2 Performance Comparison

We use the best-performing contrastive learning method, CT - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$, and the best-performing MLM method, CLS - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{MLM}}$, for comparison with existing state-of-the-art methods, namely SpotFake[30] and McCrae et al. [20]. As seen in Tables 6.1 and 6.2, all contrastive learning and MLM models outperform state-of-the-art misinformation detection methods on accuracy in both random mismatch and labeled misinformation detection. Our

Method	Accuracy	Precision	Recall	M-Precision	M-Recall
SpotFake[30]	72.05%	70.67%	74.96%	73.60%	69.17%
McCrae et al. [20]	76.40%	75.35%	78.56%	77.56%	74.24%
CLS - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{MLM}}$	81.53%	83.33%	78.82%	79.91%	84.23%
CT - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$	85.43%	85.24%	85.44%	85.62%	85.43%

Table 6.1: **Performance Comparison on Random Mismatch Detection** – Contrastive Learning method outperforms McCrae et al. in misinformation detection accuracy by 9.03%.

Method	Accuracy	Precision	Recall	M-Precision	M-Recall
SpotFake[30]	49.28%	82.04%	48.41%	18.89%	53.13%
McCrae et al. [20]	59.85%	77.86%	68.77%	18.18%	26.19%
CLS - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{MLM}}$	72.32%	82.39%	82.65%	33.73%	33.33%
CT - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$	74.81%	81.03%	88.96%	33.96%	21.43%

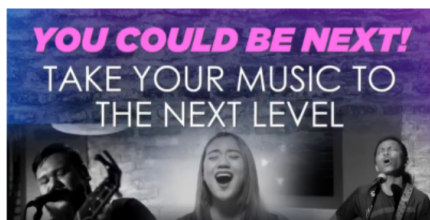
Table 6.2: **Performance Comparison on Labeled Misinformation Detection** – Contrastive Learning method outperforms McCrae et al. in misinformation detection accuracy by 14.96%.

contrastive learning method, CT - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$, outperforms McCrae et al. in misinformation detection accuracy by 9.03% on random mismatch detection and 14.96% on labeled misinformation detection.

6.3 Qualitative Examples

In Figure 6.3, we show matched and mismatched posts that are classified correctly by our the best-performing contrastive learning method, CT - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$. In Figure 6.4, we show wrong predictions made by the same model. Figure 6.4(a) shows a matched post that is classified as mismatched. In this case, the video and text provided are unrelated, and the video could be representing some higher-level symbolic meaning, which remains a challenge for vision and language understanding. Figure 6.4(b) shows a mismatched post that is classified as matched. The description of the post focuses on tickets sale, and the video shows a vote count. Likely, our model considers the numeric nature of both text and video, and classifies them as matched. Future work could investigate learning random mismatches at increasing difficulties through hard-negative mining to reduce such failures.

AUDITIONS ARE STILL OPEN! With passion comes determination. Want to take your musical talent to the next level? Make the most out of your skills and join our upcoming talent search here at Stages Sessions! Just head over to for the audition process!



(a)

When you forget your P.E Kit and it's football



(b)

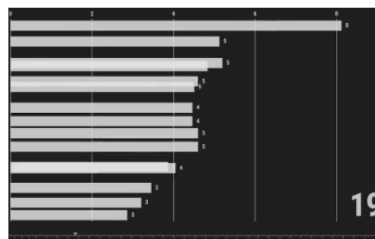
Figure 6.3: **Examples of Our Model’s Correct Predictions in Labeled Misinformation Dataset** – (a) Matched post; (b) Mismatched post.

Good to see you guys.



(a)

ANYWAY DOE, I wanted to tell y’all about my karaoke night on 7/31!! Good drinks, good smoke, good music and great vibes with yours truly Tickets on sale for \$15! Come sang with me or rap whichever is your cup of tea. Tickets are going fast!



(b)

Figure 6.4: **Examples of Our Model’s Wrong Predictions in Labeled Misinformation Dataset** – (a) Matched post; (b) Mismatched post.

6.4 Ablation Studies

In this section, we examine the way quality of text feature extraction affects detection performance, and an alternative method of two-stage training.

Text Feature Extractor	Year Published	Vocab Size	#Params	MNLI-(m/mm)[38]
BERT [8]	2018	30K	110M	84.6%/83.4%
DeBERTa-v3-Large [10]	2021	128K	304M	91.8%/91.9%

Table 6.3: **Comparison of BERT and DeBERTa** – DeBERTa is newer, larger, and performs better on MNLI than BERT.

Text Feature Extractor	Accuracy	Precision	Recall	M-Precision	M-Recall
BERT [8]	80.79%	82.01%	78.85%	79.67%	82.73%
DeBERTa-v3-Large [10]	85.43%	85.24%	85.44%	85.62%	85.43%

Table 6.4: **Effect of Text Feature Extractor Quality on Random Mismatch Detection** – DeBERTa improves accuracy by 4.64% over BERT. CT - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$ is used as the model.

Text Feature Extractor	Accuracy	Precision	Recall	M-Precision	M-Recall
BERT [8]	68.84%	81.17%	79.11%	26.67%	29.27%
DeBERTa-v3-Large [10]	74.81%	81.03%	88.96%	33.96%	21.43%

Table 6.5: **Effect of Text Feature Extractor Quality on Labeled Misinformation Detection** – DeBERTa improves accuracy by 5.97% over BERT. CT - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$ is used as the model.

6.4.1 Text Feature Extractor

We compare text features extracted from BERT[8] and DeBERTa-v3-Large[10] in Table 6.3. DeBERTa-v3 uses a similar architecture as BERT, but improves upon both the dataset it is pre-trained on, and the pre-training method. It extracts more disentangled features, as shown by its improvement in accuracy in MNLI [38]. We train using CT - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$ with each text feature extractor, and compare their performances. As shown in Tables 6.4 and 6.5, we find that using DeBERTa features improves random mismatch and labeled misinformation detection accuracy by 4.64% and 5.97%, respectively. This shows that the quality of video and text information extractors are essential to the downstream misinformation detection task.

6.4.2 Effect of Two-stage Training

During training, we combine representation learning loss and BCE loss to achieve representation learning and classification at the same time. We now examine the effect of two-stage

Method	#Stages	Accuracy	Precision	Recall	M-Precision	M-Recall
Contrastive Learning	1	85.43%	85.24%	85.44%	85.62%	85.43%
Contrastive Learning	2	84.95%	86.27%	82.87%	83.76%	87.00%
MLM	1	81.53%	83.33%	78.82%	79.91%	84.23%
MLM	2	53.36%	68.17%	12.62%	51.85%	94.11%

Table 6.6: **Effect of Two-Stage Training on Random Mismatch Detection** – Pre-training does not help misinformation detection performance.

Method	#Stages	Accuracy	Precision	Recall	M-Precision	M-Recall
Contrastive Learning	1	74.81%	81.03%	88.96%	33.96%	21.43%
Contrastive Learning	2	74.06%	82.57%	85.17%	36.49%	32.14%
MLM	1	72.32%	82.39%	82.65%	33.73%	33.33%
MLM	2	31.67%	87.72%	15.77%	22.38%	91.67%

Table 6.7: **Effect of Two-Stage Training on Labeled Misinformation Detection** – Pre-training helps contrastive learning on classifying misinformation, with higher M-Precision and higher M-Recall.

training, where we (1) learn a representation space first, and then (2) with the representation space frozen, train our model with BCE loss to learn misinformation classification using the representation space.

We train the best-performing contrastive learning method, CT - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$, and the best-performing MLM method, CLS - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{MLM}}$, again using two-stage training. Results are shown in Tables 6.6 and 6.7 for random and labeled mismatch respectively. We find that using two-stage training degrades performance for MLM significantly, and for contrastive learning slightly. We hypothesize that the representation space learned in two-stage training is not perfect, which leads to more failures in downstream detection. In particular, applying the two-stage training method is catastrophic for the MLM, because the [CLS] token is not trained properly during MLM, and that during the second stage of classification training, it fails to pay attention to the correct features for misinformation detection.

However, we notice that pre-training does improve contrastive learning’s M-Precision and M-Recall on labeled misinformation detection, as shown in Table 6.7. This means that the model is more efficient at misinformation detection, and more suitable for the task of filtering social media posts for flagging misinformation.

Method	Accuracy	Precision	Recall	M-Precision	M-Recall
SpotFake[30]	49.42%	45.50%	5.83%	49.69%	93.01%
McCrae et al. [20]	50.27%	50.18%	74.97%	50.54%	25.58%
CLS - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{MLM}}$	74.51%	71.78%	80.08%	78.04%	68.23%
CT - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$	82.26%	79.37%	87.18%	85.78%	77.34%
CT - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$, w/ 2 Stages	82.56%	81.42%	84.39%	83.80%	80.74%

Table 6.8: **Random Mismatch Experiment on COVID-19 Related Tweets** – State-of-the-art methods only achieve random-guess accuracy, while our methods greatly exceeds random-guess.

Method	Accuracy	Precision	Recall	M-Precision	M-Recall
SpotFake[30]	49.98 %	49.58%	2.96%	49.99%	96.99%
McCrae et al. [20]	49.97%	49.97%	69.72%	49.94%	30.21%
CLS - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{MLM}}$	58.25%	55.32%	85.67%	68.26%	30.82%
CT - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$	67.37%	64.99%	75.27%	70.63%	59.46%
CT - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$, w/ 2 Stages	68.45%	69.63%	65.42%	67.39%	71.47%

Table 6.9: **Random Mismatch Experiment on Russia-Ukraine Related Tweets** – Pre-training improves model accuracy on unseen test distribution.

6.5 Performance Comparison on Topic-Specific Random Mismatch

We also test different methods’ performance on video-text random mismatches on a specific topic, namely COVID-19 or Russia-Ukraine crisis. Results are shown in Tables 6.8 and 6.9 for random mismatch of COVID-19 and Russia-Ukraine crisis respectively. For COVID-19, we select 1,646 tweets that contain COVID-related terms from the test set of 1 million Twitter dataset. For Russia-Ukraine crisis, since the dataset’s collection timeframe barely overlaps with the crisis’s timeline, we recollect 60,000 twitter posts of Russia-Ukraine related terms. We conduct the same data cleaning procedures in Section 6.1, and randomly select 10,000 posts for testing. For both datasets, we consider given tweets as positive, matched tweets, and create one mismatched sample for each video by selecting a random text within the same dataset.

In both COVID and Russia-Ukraine experiments, we see that previous state-of-the-art methods only achieve about accuracy of a random-guess, 50%. We speculate that these methods only learn to detect topic-mismatch, where video and text are on unrelated topics.

Thus, they do not perform well on one-topic random mismatch testing. Our methods outperform state-of-the-art methods by 32.29% for COVID-19 and 18.47% for the Russia-Ukraine datasets, perhaps implying that they understand the fine-grain details in video and text, rather than only inferring using the general topic in video and text.

We see that the best performance on random mismatch topic specific datasets drops by 2.87% and 16.98% for COVID-19 and Russia-Ukraine respectively as compared to all tweets. As shown in Table 6.8, COVID-19’s M-Recall, the probability of detecting a true mismatch, drops by 6.26% as compared to all tweets. This drop could be caused by the increasing difficulty of topic specific mismatches. The large performance drop in Russia-Ukraine random mismatch is likely caused by the smaller training dataset for Russia-Ukraine as compared to COVID-19. Clearly, larger training data results in better learned video and text representations and hence better detection accuracy.

We also observe that CT - $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{NCE}}$ with two-stage training performs slightly better than 1-stage training for both COVID-19 and Russia-Ukraine random mismatch. COVID-19 and Russia-Ukraine datasets are unseen test distributions, and are very different from the used training data. We hypothesize that pre-training learns a more robust representation towards test-time distribution shifts, thus performing better in topic-specific random mismatch detection.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

Misinformation spread in social media video posts can cause chaos in times of national or global events. To combat misinformation, we propose to use effective representation learning methods to improve accuracy in misinformation detection. Our Contrastive Learning and MLM methods both outperform state-of-the-art methods, and our best-performing Contrastive Learning method achieves accuracy 9.03% higher than McCrae et al. [20] in random mismatch detection, and 14.96% higher in a labeled misinformation dataset of 401 samples. We show that representation learning improves misinformation detection accuracy, and preserving temporal information during fusion is important. We further discuss through ablation studies that better feature extractor leads to better detection accuracy, and two-stage, pre-train then fine-tune, training method, does not perform better than one-stage training, but has better potential for misinformation detection in the wild.

7.2 Future Work

There are many directions for future work. (1) Improve representation learning by combining MLM with contrastive learning. Our preliminary work shows that model tends to "shortcut" the MLM loss when it is trained together with NCE loss, where MLM loss quickly converges to a near-zero value. (2) Improve unsupervised grounding model's accuracy. In Section 5.4, we demonstrate the potential of unsupervised method in grounding misinformation. Current inference methods are biased towards probability of text without video, because of how token probability is inferred - each individual token is masked and then inferred, rather than word-by-word. Masking word-by-word may improve the balance between token distributions with and without video. (3) In MLM, it is possible to use transformer's attention mechanism to track which video segments and text tokens that are being paid most attention to by the [CLS] token when making a decision. This can potentially provide explainability to the decision as it guides the decision to video/text of interest. (4) Currently, our model

learns all mismatches equally, but because of the way random mismatches are generated, it is possible to learn random mismatches at increasing difficulties through hard-negative mining to reduce the false detection of misinformation. (5) There is no public video-text misinformation detection dataset available. With our system setup in Section 6.1, labeled misinformation collection can be easily scaled and deployed to crowd-sourcing platforms.

7.3 Engineering Traps to Avoid in Vision-and-Language Research

Last but not least, there are many "traps" that I have fallen into and wasted a lot of time and effort on. I would like to list three major ones here for future references. (1) Remove "retweets" in data. "Retweets" are simply one forwarding a tweet to one's account without any edit. This results in duplication of tweets in the dataset and potential test set leakage. If one finds that model training is not overfitting, it is either because the model is too small, or there is test set leakage. Therefore, data cleaning is important. Do exhaustive data exploratory analysis before diving into modeling. (2) Modeling learning to classify mismatch through padding mask. Because of the variable lengths of video and text, and transformer only accepting fixed lengths of inputs, we always have to pad inputs to the maximum length of inputs within the dataset. During mismatch generation, we need to swap both text and its corresponding padding mask, because the model is very good at detecting if the padding mask and text are matching to make a good classification. (3) Library versioning. This one has wasted a lot of GPU hours. We found that running the same code can result in more than 1% difference in accuracy. We have seeded all random processes so the code is deterministic. As we found out, a newer version of PyTorch Lightning, 1.5.10, results in 1% lower performance. Therefore, it is important to keep library versions consistent across experiments.

Bibliography

- [1] Hassan Akbari et al. “Vatt: Transformers for multimodal self-supervised learning from raw video, audio and text”. In: *arXiv preprint arXiv:2104.11178* (2021).
- [2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. *Layer Normalization*. 2016. DOI: [10.48550/ARXIV.1607.06450](https://doi.org/10.48550/ARXIV.1607.06450). URL: <https://arxiv.org/abs/1607.06450>.
- [3] Michael Barthel, Amy Mitchell, and Jesse Holcomb. “Many Americans Believe Fake News Is Sowing Confusion”. In: *Pew Research Center* (Dec. 2016).
- [4] Ting Chen et al. “A Simple Framework for Contrastive Learning of Visual Representations”. In: *arXiv preprint arXiv:2002.05709* (2020).
- [5] Xinlei Chen et al. *Improved Baselines with Momentum Contrastive Learning*. 2020. DOI: [10.48550/ARXIV.2003.04297](https://doi.org/10.48550/ARXIV.2003.04297). URL: <https://arxiv.org/abs/2003.04297>.
- [6] S. Chopra, R. Hadsell, and Y. LeCun. “Learning a similarity metric discriminatively, with application to face verification”. In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*. Vol. 1. 2005, 539–546 vol. 1. DOI: [10.1109/CVPR.2005.202](https://doi.org/10.1109/CVPR.2005.202).
- [7] P. Das et al. “A Thousand Frames in Just a Few Words: Lingual Description of Videos through Latent Topics and Sparse Object Stitching”. In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*. 2013. URL: http://web.eecs.umich.edu/~jjcorso/pubs/jcorso_CVPR2013_dpmtm.pdf.
- [8] Jacob Devlin et al. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805* (2018).
- [9] Yi Fung et al. “InfoSurgeon: Cross-Media Fine-grained Information Consistency Checking for Fake News Detection”. In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Online: Association for Computational Linguistics, Aug. 2021, pp. 1683–1698. DOI: [10.18653/v1/2021.acl-long.133](https://doi.org/10.18653/v1/2021.acl-long.133). URL: <https://aclanthology.org/2021.acl-long.133>.
- [10] Pengcheng He, Jianfeng Gao, and Weizhu Chen. *DeBERTaV3: Improving DeBERTa using ELECTRA-Style Pre-Training with Gradient-Disentangled Embedding Sharing*. 2021. arXiv: [2111.09543](https://arxiv.org/abs/2111.09543) [cs.CL].

- [11] Yonghun Jang, Chang-Hyeon Park, and Yeong-Seok Seo. “Fake News Analysis Modeling Using Quote Retweet”. In: *Electronics* 8.12 (2019). ISSN: 2079-9292. DOI: [10.3390/electronics8121377](https://doi.org/10.3390/electronics8121377). URL: <https://www.mdpi.com/2079-9292/8/12/1377>.
- [12] Jie Lei et al. “Less is More: ClipBERT for Video-and-Language Learning via Sparse Sampling”. In: *CVPR*. 2021.
- [13] Jie Lei et al. “TVQA: Localized, Compositional Video Question Answering”. In: *EMNLP*. 2018.
- [14] Linjie Li et al. *HERO: Hierarchical Encoder for Video+Language Omni-representation Pre-training*. 2020. DOI: [10.48550/ARXIV.2005.00200](https://doi.org/10.48550/ARXIV.2005.00200). URL: <https://arxiv.org/abs/2005.00200>.
- [15] Jingzhou Liu et al. “Violin: A Large-Scale Dataset for Video-and-Language Inference”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2020.
- [16] Sahil Loomba et al. “Measuring the impact of COVID-19 vaccine misinformation on vaccination intent in the UK and USA”. In: *Nature Human Behaviour* 5.3 (Mar. 2021), pp. 337–348. ISSN: 2397-3374. DOI: [10.1038/s41562-021-01056-1](https://doi.org/10.1038/s41562-021-01056-1). URL: <https://doi.org/10.1038/s41562-021-01056-1>.
- [17] Grace Luo, Trevor Darrell, and Anna Rohrbach. “NewsCLIPpings: Automatic Generation of Out-of-Context Multimodal Media”. In: *CoRR* abs/2104.05893 (2021). arXiv: [2104.05893](https://arxiv.org/abs/2104.05893). URL: <https://arxiv.org/abs/2104.05893>.
- [18] Huaishao Luo et al. “UniVL: A Unified Video and Language Pre-Training Model for Multimodal Understanding and Generation”. In: *arXiv preprint arXiv:2002.06353* (2020).
- [19] Mohit Mayank, Shakshi Sharma, and Rajesh Sharma. “DEAP-FAKED: Knowledge Graph based Approach for Fake News Detection”. In: *CoRR* abs/2107.10648 (2021). arXiv: [2107.10648](https://arxiv.org/abs/2107.10648). URL: <https://arxiv.org/abs/2107.10648>.
- [20] Scott McCrae, Kehan Wang, and Avidesh Zakhori. *Multi-Modal Semantic Inconsistency Detection in Social Media News Posts*. 2021. arXiv: [2105.12855](https://arxiv.org/abs/2105.12855) [cs.CV].
- [21] Antoine Miech et al. *End-to-End Learning of Visual Representations from Uncurated Instructional Videos*. 2019. DOI: [10.48550/ARXIV.1912.06430](https://doi.org/10.48550/ARXIV.1912.06430). URL: <https://arxiv.org/abs/1912.06430>.
- [22] Antoine Miech et al. “End-to-End Learning of Visual Representations from Uncurated Instructional Videos”. In: *CVPR*. 2020.
- [23] Antoine Miech et al. “HowTo100M: Learning a Text-Video Embedding by Watching Hundred Million Narrated Video Clips”. In: *ICCV*. 2019.
- [24] Patricia Moravec, Randall Minas, and Alan Dennis. “Fake News on Social Media: People Believe What They Want to Believe When it Makes No Sense at All”. In: *SSRN Electronic Journal* (Jan. 2018). DOI: [10.2139/ssrn.3269541](https://doi.org/10.2139/ssrn.3269541).

- [25] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. *Representation Learning with Contrastive Predictive Coding*. 2018. DOI: [10.48550/ARXIV.1807.03748](https://doi.org/10.48550/ARXIV.1807.03748). URL: <https://arxiv.org/abs/1807.03748>.
- [26] Jeff Z. Pan et al. “Content Based Fake News Detection Using Knowledge Graphs”. In: *SEMWEB*. 2018.
- [27] Olga Papadopoulou et al. “A corpus of debunked and verified user-generated videos”. In: *Online Information Review* (2018). DOI: [10.1108/OIR-03-2018-0101](https://doi.org/10.1108/OIR-03-2018-0101).
- [28] Alec Radford et al. “Learning Transferable Visual Models From Natural Language Supervision”. In: *CoRR* abs/2103.00020 (2021). arXiv: [2103.00020](https://arxiv.org/abs/2103.00020). URL: <https://arxiv.org/abs/2103.00020>.
- [29] Lanyu Shang et al. “A Multimodal Misinformation Detector for COVID-19 Short Videos on TikTok”. In: *2021 IEEE International Conference on Big Data (Big Data)*. 2021, pp. 899–908. DOI: [10.1109/BigData52589.2021.9671928](https://doi.org/10.1109/BigData52589.2021.9671928).
- [30] Shivangi Singhal et al. “SpotFake: A Multi-modal Framework for Fake News Detection”. In: *2019 IEEE Fifth International Conference on Multimedia Big Data (BigMM)*. 2019, pp. 39–47. DOI: [10.1109/BigMM.2019.00-44](https://doi.org/10.1109/BigMM.2019.00-44).
- [31] Chen Sun et al. “Videobert: A joint model for video and language representation learning”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 7464–7473.
- [32] S Shyam Sundar, Maria D Molina, and Eugene Cho. “Seeing Is Believing: Is Video Modality More Powerful in Spreading Fake News via Online Messaging Apps?” In: *Journal of Computer-Mediated Communication* 26.6 (Aug. 2021), pp. 301–319. ISSN: 1083-6101. DOI: [10.1093/jcmc/zmab010](https://doi.org/10.1093/jcmc/zmab010). eprint: <https://academic.oup.com/jcmc/article-pdf/26/6/301/41139661/zmab010.pdf>. URL: <https://doi.org/10.1093/jcmc/zmab010>.
- [33] Reuben Tan, Bryan Plummer, and Kate Saenko. “Detecting Cross-Modal Inconsistency to Defend Against Neural Fake News”. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, Nov. 2020, pp. 2081–2106. DOI: [10.18653/v1/2020.emnlp-main.163](https://doi.org/10.18653/v1/2020.emnlp-main.163). URL: <https://aclanthology.org/2020.emnlp-main.163>.
- [34] Yansong Tang et al. “COIN: A Large-scale Dataset for Comprehensive Instructional Video Analysis”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019.
- [35] Makarand Tapaswi et al. “MovieQA: Understanding Stories in Movies through Question-Answering”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.
- [36] Ashish Vaswani et al. *Attention Is All You Need*. 2017. arXiv: [1706.03762](https://arxiv.org/abs/1706.03762) [cs.CL].

- [37] Soroush Vosoughi, Deb Roy, and Sinan Aral. “The spread of true and false news online”. In: *Science* 359.6380 (2018), pp. 1146–1151. DOI: [10.1126/science.aap9559](https://doi.org/10.1126/science.aap9559). eprint: <https://www.science.org/doi/pdf/10.1126/science.aap9559>. URL: <https://www.science.org/doi/abs/10.1126/science.aap9559>.
- [38] Adina Williams, Nikita Nangia, and Samuel Bowman. “A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference”. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, 2018, pp. 1112–1122. URL: <http://aclweb.org/anthology/N18-1101>.
- [39] Yonghui Wu et al. *Google’s Neural Machine Translation System: Bridging the Gap between Human and Machine Translation*. 2016. arXiv: [1609.08144](https://arxiv.org/abs/1609.08144) [cs.CL].
- [40] Hu Xu et al. *VideoCLIP: Contrastive Pre-training for Zero-shot Video-Text Understanding*. 2021. DOI: [10.48550/ARXIV.2109.14084](https://doi.org/10.48550/ARXIV.2109.14084). URL: <https://arxiv.org/abs/2109.14084>.
- [41] Jun Xu et al. “MSR-VTT: A Large Video Description Dataset for Bridging Video and Language”. In: IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), June 2016. URL: <https://www.microsoft.com/en-us/research/publication/msr-vtt-a-large-video-description-dataset-for-bridging-video-and-language/>.
- [42] Zhou Yu et al. “ActivityNet-QA: A Dataset for Understanding Complex Web Videos via Question Answering”. In: *AAAI*. 2019, pp. 9127–9134.
- [43] Luwei Zhou, Chenliang Xu, and Jason J Corso. “Towards Automatic Learning of Procedures From Web Instructional Videos”. In: *AAAI Conference on Artificial Intelligence*. 2018, pp. 7590–7598. URL: <https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17344>.
- [44] Dimitri Zhukov et al. “Cross-task weakly supervised learning from instructional videos”. In: 2019.