CHECKER: Detecting Clickbait Thumbnails with Weak Supervision and Co-teaching

Tianyi Xie[1⋆], Thai Le[2], and Dongwon Lee[2]

[1] Shanghai Jiao Tong University, China, tianyixie77@gmail.com
[2] The Pennsylvania State University, USA, {tql3,dongwon}@psu.edu

Abstract. Clickbait thumbnails on video-sharing platforms (e.g., YouTube, Dailymotion) are small catchy images that are designed to entice users to click to view the linked videos. Despite their usefulness, the landing videos after click are often inconsistent with what the thumbnails have advertised, causing poor user experience and undermining the reputation of the platforms. In this work, therefore, we aim to develop a computational solution, named as CHECKER, to detect clickbait thumbnails with high accuracy. Due to the fuzziness in the definition of clickbait thumbnails and subsequent challenges in creating high-quality labeled samples, the industry has not coped with clickbait thumbnails adequately. To address this challenge, CHECKER shares a novel clickbait thumbnail dataset and codebase with the industry, and exploits: (1) the *weak supervision* framework to generate many noisy-but-useful labels, and (2) the *co-teaching* framework to learn robustly using such noisy labels.Moreover, we also investigate how to detect clickbaits on video-sharing platforms with both thumbnails and titles, and exploit recent advances in vision-language models. In the empirical validation, CHECKER outperforms five baselines by at least 6.4% in F1-score and 4.2% in AUC-ROC. The codebase and dataset from our paper are available at: https://github.com/XPandora/CHECKER.

Keywords: Clickbait thumbnail · Weak supervision · Co-teaching · Learning with noisy labels

1 Introduction

In recent a few years, the popularity of video-sharing platforms (e.g., YouTube, Dailymotion, and Vimeo) has dramatically increased. According to the recent survey by Pew Research[1], for instance, around three-quarters of U.S. adults (73%) use YouTube, surpassing 69% of U.S. adults using Facebook. As such, it is a critically important problem for such platforms to maintain a clean ecosystem and provide pleasant experience to users. However, one phenomenon severely polluting this ecosystem is the prevalence of the so-called **clickbait thumbnails**, small catchy images that are designed to entice users to click to view the linked videos

⋆ Part of the work was done while the author visited Penn State during the summer of 2019 as an intern.

[1] http://tiny.cc/3jkvtz

(e.g., several examples shown in Figure 1). Such clickbait thumbnails are often deceptive, sensationalized, exaggerating, or misleading, sometimes accompanied by eye-catching titles. The emergence of thumbnails is partially due to the desire of content creators to increase the view counts for diverse reasons (e.g., monetary gain). Despite their attractiveness at first glance, however, the landing videos may have the contents different from what the thumbnails have advertised. Such inconsistency then leads to users' unpleasant online experience and deteriorates the reputation of video-sharing platforms.

One trivial solution to combat clickbait thumbnails is to employ human annotators to review and tag clickbait thumbnails. However, not only it is costly, but also it cannot scale well to match the sheer volume of videos uploaded on popular video-sharing platforms, calling for computational and scalable solutions. Therefore, to mitigate this phenomenon of clickbait thumbnails on video-sharing platforms, the aim of this work to develop a machine learning based solution that can detect clickbait thumbnails with a high accuracy. Despite the closely related problem of detecting (text-based) clickbait news headlines has been well studied (e.g., [24, 9, 6]), the detection of clickbait thumbnails has been relatively less explored and existing solutions (e.g., [28, 23]) are based on impractical settings or show unsatisfactory accuracies. Moreover, solving the problem of detecting clickbait thumbnails using machine learning framework needs to cope with a few inherent challenges:

- Due to the subjective and ambiguous nature in the definition of clickbait thumbnails, it is non-trivial to build a clean supervised learning environment with ample labeled samples. As the tolerance levels of people often differ, a clearly annoying clickbait thumbnail to A can be perfectly entertaining thumbnail to B. Even if one uses human annotators to tag clickbait thumbnails, it is unclear what specific instruction one has to give to the annotators.
- As such, achieving consensus on a single clickbait thumbnail among multiple human annotators is challenging (and costly). Further, even after consensus, human annotated labels for clickbait thumbnails can be noisy.
- Finally, achieving high detection accuracy using rich features found in various meta-data of landing videos may not be a practical solution (e.g., [28, 23]). This is because in real settings, users are often given only a pair of information (i.e., thumbnail and title) to determine to click or not. Therefore, an ideal solution is to mimic the situation and detect clickbait thumbnails using multi-modal features from the pair of thumbnail and title.

In an attempt to address the aforementioned challenges in detecting clickbait thumbnails, this paper presents CHECKER (Clickbait tHumbnail dEtection with Co-teaching and weaK supERvision), which leverages weak supervision to generate noisy-but-useful labels and adopts co-teaching [13] to learn robustly from such noisy labels. In addition, different from prior works [28, 23], we are interested in detecting clickbait thumbnails using only the pair of a thumbnail and title, which simulates the real users' experience while browsing video-sharing platforms and avoids the cold start problem when statistics of a new video is not available. To this end, we first collect 8,987 videos along with their metadata from YouTube,

Fig. 1. Examples of clickbait thumbnails. Though they are eve-catching at first glance, the content of the linked videos is inconsistent with what these thumbnails have advertised.

including the thumbnail and title. Note that the collected metadata of video are used to generate noisy labels, but will not be used in either training or inference. Then, we collect the initial labels for a small subset of these thumbnails via crowdsourcing on the Amazon Mechanical Turk platform. Note that most of the thumbnails remain unlabeled. To make a full use of these unlabeled thumbnails, then, we adopt the weak supervision framework and generate noisy-but-useful labels for them. Then, to prevent the powerful neural networks (NNs) from memorizing these noisy labels (thus degrading accuracy), we furthermore adopt the co-teaching strategy [13] to filter out thumbnails with wrong labels while training. By and large, our main contributions are as follows:

- We release a clickbait thumbnail detection dataset, which consists of 8,987 videos with their metadata from YouTube, and 787 of them get labeled through crowdsourcing.
- We propose CHECKER for clickbait thumbnail detection, which leverages weak supervision to generate labels for thumbnails with over 80% accuracy. Specifically, based on the characteristics of clickbait thumbnails, we design several useful labeling functions as weak supervision sources and then combine them to generate labels. Furthermore, co-teaching strategy is also applied in the training to cope with the noise among generated labels.
- We exploit recent advances in vision-language models and make a comprehensive comparison. Moreover, extensive experiments are conducted to show that our method effectively alleviates the issue of high-quality labeled training data shortage in training clickbait thumbnail detectors.

2 Related Work

2.1 Clickbait Headline Detection

There is a growing interest in studying misinformation on social media. One line of research focuses on the detection of clickbait headlines. Online content creators use these clickbait titles to attract attention and lure visitors to click on a hyperlink of a target landing web page [9], which may contain misinformation. Thus, clickbait headlines have become a popular medium for mass propagation of false news. To

explore what makes a headline "clickbaity", [17] conduct three clickbait studies. To effectively detect clickbait headlines, most of existing approaches train machine learning (ML) detectors with features that are either carefully engineered [6, 5, 10] or automatically learnt via deep NNs [22, 1]. Moreover, [24, 26, 15] further improves those detectors by augmenting their training dataset with synthetic clickbait headlines. In this work, we turn to study another type of clickbait but deserve more attention in the current literature: *clickbait thumbnail*.

2.2 Clickbait Thumbnail Detection

Clickbait thumbnails are small catchy images that are designed to entice users to click to view a particular video, with a defining characteristic of being deceptive, sensationalized, exaggerating, or misleading. Compared to clickbait headlines, only a few pioneering works start to study these misleading thumbnails. To the best of our knowledge, [28] first studies the clickbait problem on Youtube and builds a VAE-based model for automatic detection. [23] proposes a content-agnostic approach to detect clickbait videos, which mainly makes use of the comments of videos. In spite of their progress, both of them suffer from the shortage of a reliable training corpus. [19] also indicates that automatic clickbait detection on YouTube is still far out of reach due to the paucity of training data. To deal with the lack of available datasets, [28] retrieves videos from clickbait and non-clickbait channels, and obtain labels for videos based on the label (clickbait or non-clickbait) of the channels they belong to. However, this approach is not convincing since even non-clickbait channels may publish clickbait videos. [23] also constructs a dataset of 625 videos, but such size is usually too small to train a robust deep neural network. Hence, in this paper, we make further efforts to tackle the shortage of training samples in clickbait thumbnail detection.

2.3 Vision-Language Model

Various vision-language tasks have attracted the attention of the research community in recent years, such as Image Captioning and Visual Question Answer, which require the capability to understand and fuse multimodal features. Early works in vision and language understanding usually design separate models for each modality followed by a multi-modal fusion layer. In this case, bi-linear fusion is thought to be more expressive but tends to result in an excess of parameters. Subsequent work address this issue through low-rank decomposition [12, 3, 4]. In addition, more recent works show that a joint pre-training over both modalities enables the model to easily adapt to downstream tasks. Some work therefore train a holistic network on a large training corpus, which is able to give a joint embedding of vision and language, such as VisualBERT [16], LXMERT [25] and UNITER [8]. In this work, we apply and compare these state-of-the-art methods and models to the clickbait thumbnail detection task.

Table 1. The overview of our clickbait thumbnail dataset. As we can see, even clickbait channels may use non-clickbait thumbnails, and the same is with non-clickbait channels.

		clickbait channel	non-clickbait channel	total
Train	# clickbait thumbnail	146	38	184
	# non-clickbait thumbnail	150	256	406
	# unlabeled thumbnail	3851	4349	8200
Test	# clickbait thumbnail	49	15	64
	# non-clickbait thumbnail	45	88	133
	# unlabeled thumbnail	-	-	-

3 Building Dataset

In this work, we aim to study the clickbait thumbnail detection problem on YouTube. Since there is not any reliable dataset of clickbait thumbnails in the literature, we first need to collect a high-quality labeled dataset for our study. Our data collection process includes two steps: (i) data acquisition and (ii) label collection.

3.1 Data Acquisition

There are many more videos with benign than with clickbait thumbnails. Due to this imbalanced nature between clickbaits and non-clickbaits, collecting data points randomly from video-sharing platforms will result in a dataset with a highly skewed class distribution. Thanks to prior work [28], we first retrieve a list of clickbait and non-clickbait channels on YouTube. By leveraging YouTube Data API [2], we crawl 8,987 videos as well as the metadata from these channels, which are published between May and July of 2019. Note that here we use the video's source as an approximation for its clickbaitness and we also try to collect the same amount of videos from each channel to prevent uneven data distribution.

Generally, the metadata can be categorized into four groups: (1) title and description; (2) thumbnail; (3) statistics (e.g., like and dislike count, etc.); (4) comments. Particularly, assuming that popular comments represent the opinion of the majority, we select only the top 10 comments with the highest like count for each video.

3.2 Label Collection

Though we have collected a large number of data from YouTube, all of them are still unlabeled. For the sake of model evaluation, ground truth labels are indispensable. However, due to the vague and ambiguous definition of clickbait thumbnails, it is impossible to annotate all of them in a short time. To collect high-quality labels, we first define clickbait thumbnail as follows:

[2] `https://developers.google.com/youtube/v3`

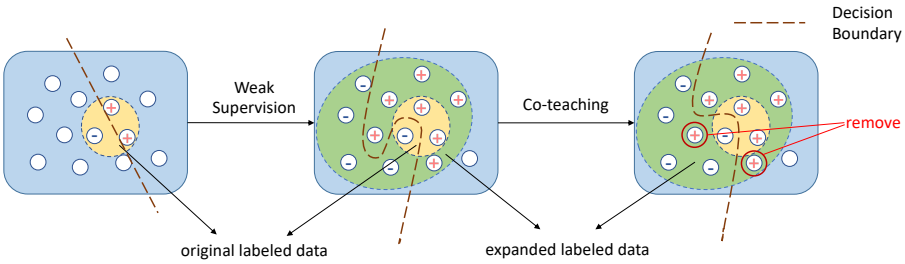

Fig. 2. The overview of data flow in CHECKER. The decision boundary will change along with the distribution of training data.

Definition: Clickbait Thumbnail. Clickbait thumbnail is a thumbnail that is inconsistent with the gist of the corresponding video that it represents.

Based on this definition, we publish labeling tasks on the Amazon Mechanical Turk platform and utilize crowdsourcing to label parts of samples in the dataset. To simulate the experience when users are browsing video-sharing portals, we ask workers to first inspect the thumbnail and title of a video. Then workers are required to watch the video for at least one minute to grab the gist. By comparing the content of the video to the meaning conveyed by the thumbnail and title, workers should be able to tell whether the thumbnail is a clickbait.

To ensure the quality of labels, for each sample, we invite 5 workers to label and use the majority vote to determine the final label. Finally, 787 samples get labeled through crowdsourcing. For experiments, we take 197 of them as the test set while others as a part of the training set. Table 1 provides an overview of our collected dataset.

4 The Proposed Method: CHECKER

Our objective is to train a discriminative model with partially labeled training data. In this paper, we present our framework CHECKER, which takes the advantage of both weak supervision and co-teaching. Basically, our framework can be split into two stages: **generating noisy labels** and **learning from noisy labels**. Specifically, we leverage weak supervision to generate noisy labels while adopt the co-teaching algorithm to remove samples with wrong labels in learning from noisy labels. Figure 2 presents the basic data flow in our framework.

4.1 Generating Noisy Labels

Though we have collected some labels through crowdsourcing, a large number of samples are still unlabeled. To make these unlabeled thumbnails available for training models, we leverage the weak supervision to generate labels for them. Specifically, weak supervision means noisy, limited, or imprecise sources

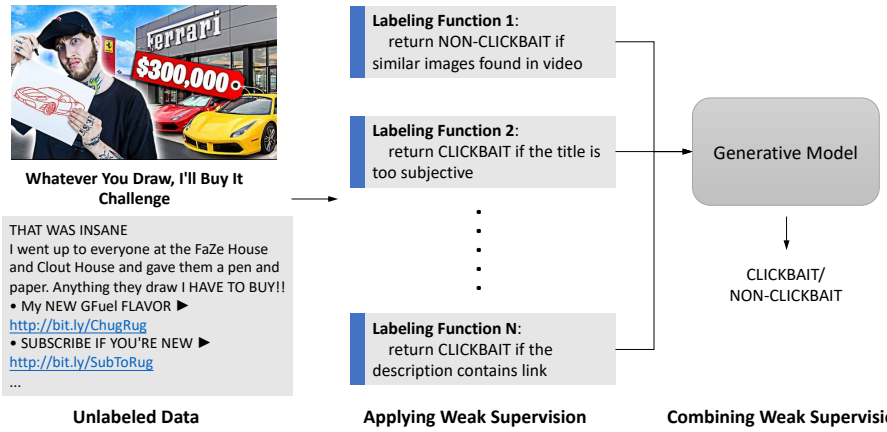

Fig. 3. The overview of generating labels. Specifically, we first design labeling functions as weak supervision sources and then use a generative model to combine them to produce the final label.

are used to provide supervision signals for labeling large amounts of training data. These cheap labels can be obtained through a set of simple rules instead of manual annotation. This approach, to a great extent, releases researchers from spending too much time in acquiring high-quality labels. In the clickbait thumbnail detection task, weak supervision sources can be various labeling functions based on the characteristics of the thumbnail. For instance, the presence of the word 'clickbait' in the comments of a video on Youtube indicates that this video's thumbnail may be a clickbait.

Here we explain why weak supervision is suitable for generating labels for clickbait thumbnail detection. First, though there is no explicit definition for clickbait thumbnail that enables us to label data quickly, we can easily speak out several rules to roughly judge whether the thumbnail is a clickbait. One simple rule can be that if the thumbnail is one frame of the video, then it should be a non-clickbait thumbnail since it does truthfully reflect the content of this video. Such rules can be regarded as weak supervision sources and are easy to implement. Second, correctly identifying clickbait thumbnails requires people to fully understand the video content and then compare it with the thumbnail, which is extremely time-consuming, while utilizing weak supervision can prevent such heavy work. Third, since we can get various weak supervision sources by designing different labeling rules, combining them as an ensemble enables us to obtain high-quality labels.

Once proper labeling functions are designed, the critical problem becomes how to regulate and utilize these results. Recent advances [20, 11] in weak supervision have already made some breakthroughs with regard to this problem, which usually builds a generative model to estimate accuracy and correlations of weak supervision sources.

Design Labeling Functions. Intuitively, the quality of the final generated labels is positively correlated to the quality of labeling functions. Hence, it is crucial to design labeling functions as high quality as possible, though in most cases there does not exist a single perfect labeling function. Besides, the diversity as well as the coverage of labeling functions should also be considered. In other words, different labeling functions should focus on different features to prevent bias, and in the meantime, they should assign labels to as many samples as possible.

To formalize, each weak supervision λ_j works as follows:

$$\tilde{y}_{ij} = \lambda_j(x_i), \tag{1}$$

where x_i denotes the feature of i-th data sample, including title x_i^{ti}, thumbnail x_i^{th}, description x_i^d, video x_i^v, statistics x_i^s and comment x_i^c, and $\tilde{y}_{ij} \in \{-1, 0, 1\}$ denotes the labeling result given by j-th labeling function. Note that '-1' refers to abstain, '0' refers to non-clickbait while '1' refers to clickbait.

Based on the characteristic of clickbait thumbnails, we design labeling functions according to the following aspects:

- *Channel.* [28] once used the label of the channel for the videos inside. Though this is actually not corrected, the label of channels indeed indicates the general property of thumbnails. As shown in Table 1, most of clickbait thumbnails are from clickbait channels while non-clickbait channels seldom upload clickbait thumbnails. Hence, we adopt the label of the channel as one labeling function.
- *Thumbnail.* As shown in Figure 1, One main critical feature of clickbait thumbnail is the presence of those striking texts, which are artificially added by video uploaders. To draw the attention of users, such text usually occupies a large space of a thumbnail. We therefore employ the optical character recognition (OCR) service [3] to measure the ratio of the text area to the whole image. With a proper threshold, a thumbnail whose text area exceeds the threshold value can be categorized as clickbait. In addition, since telling whether a thumbnail is a clickbait needs comparison with video content, we also adopt dHash algorithm h [4] to calculate the similarity between the thumbnail and frames of the video. Specifically, we calculate the L1 distance between the dHash code of the thumbnail and that of each frame, and the similarity score is the minimum value among all the distances. To formulize, the similarity score is calculated as follows:

$$d_{in} = \left\| h(x_i^{th}) - h(x_{in}^v) \right\|_1, \tag{2}$$

$$s_i = \min \{d_{i1}, d_{i2}, ..., d_{iN}\}, \tag{3}$$

  where x_{in}^v denotes the n-th frame of the video and N is the frame number. d_{in} denotes the L1 distance between the thumbnail and n-th frame while s_i

[3] https://cloud.google.com/vision/docs/ocr

[4] http://www.hackerfactor.com/blog/index.php?/archives/529-Kind-of-Like-That.html

Table 2. Statistics of each labeling function on labeled data. Note that polarity represents the set of labels that labeling functions will output and '1' refers to clickbait while '0' refers to non-clickbait.

Labeling Function	Polarity	Coverage	Overlaps	Conflicts	Correct	Incorrect	Acc.
channel&thumbnail-based	1	0.202	0.108	0.089	110	49	0.692
channel&statistics-based	1	0.088	0.067	0.048	45	24	0.652
channel-based	0	0.495	0.348	0	364	26	0.933
title-based	0	0.492	0.411	0.105	295	93	0.760
thumbnail-based	0	0.131	0.119	0.016	95	8	0.922
description-based	0	0.084	0.079	0.002	59	7	0.894

denotes the similarity score between the thumbnail and the video. A high similarly score means that the thumbnail indeed reflect the video content, which indicates a benign thumbnail.

- *Title.* Clickbait thumbnails are usually presented with eye-catching titles. Generally, to catch users' attention, exaggerated titles tend to exhibit strong subjectivity. On the other hand, a title with high subjectivity indicates a great possibility of clickbait. Therefore we also use TextBlob [5] to mine the deep semantics behind the title and consider those with high subjective scores as clickbait.
- *Description.* Clickbait on video-sharing platforms usually displays links to other websites in the description for the purpose of advertising. Thus, according to whether the link exists in the description, we can judge the class of thumbnails.
- *Statistics.* Statistics includes like count, dislike count, view count and comment count. Generally, users tend to close the video webpage without leaving comments once they discover it's a clickbait. Hence, we consider the video with a low comment to view ratio as a potential clickbait.

Based on the above observation, we write 6 labeling functions ($\lambda_1, ..., \lambda_N$, where $N = 6$). Performance of labeling functions on labeled data is provided in Table 2 and their detailed implementation can be found in the provided codebase.

Combining Labeling Results. By applying all labeling functions to all unlabeled data, we obtain a label matrix Λ, where $\Lambda_{i,j} = \lambda_j(x_i)$. To combine the different labeling results, we are essentially aiming to build a generative model G that functions as follows:

$$\tilde{y}_i = G(\Lambda_i), \tag{4}$$

where Λ_i refers to the labeling result of all weak supervision sources for the unlabeled data sample x_i.

We evaluate and compare three different generative models on the labeled data. The comparison results is presented in Table 3. Note that for Snorkel [20]

[5] https://github.com/sloria/textblob

Table 3. Comparison of different generative model.

Method	Accuracy	F1 score	Precision	Recall
Majority Voter	0.836	0.635	0.717	0.570
Epoxy [7]	0.784	0.637	0.638	0.635
Snorkel [20]	0.808	0.667	0.670	0.663

and Epoxy [7], we train them with unlabeled data before evaluation. Based on the comparison result, we adopt the majority voter for further experiments for two reasons: (1) the accuracy of majority voter is higher so that there is less noise among the generated labels, and (2) considering that clickbait training samples are more important due to its paucity in our dataset, a higher precision means more high-quality 'clickbait' labels, which enables the model to learn a better decision boundary. Using this generative model, we generate labels for 7,039 unlabeled data in total while 1,061 samples remain unlabeled since none of labeling functions assigns labels for them. After label generation, the size of our labeled training samples has increased to 7,630.

4.2 Learning from Noisy Labels

The objective of this stage is to train a robust vision-language classifier with the generated labels. Specifically, given a thumbnail x_i^{th} and title x_i^{ti}, the task of this classifier is to predict a label $\hat{y}_i$ indicating whether it is a clickbait. Plus, though we have obtained a large number of labels with weak supervision, these generated labels are noisy. Note that noisy labels mean that not all labels are correct. It is known that the strong fitting capability of machine learning models such as neural networks may lead itself to overfit the noise, which would finally result in a poor generalization. Hence, for robust learning, it is also critical to combat noisy labels during training.

Model Architecture. Since the clickbait thumbnail detection is a vision-language task, we exploit recent advances in vision-language areas to build a clickbait detector, as shown on the left side of Figure 4. Specifically, we use the ResNet-50 model [14] pre-trained on ImageNet to extract the image embedding while adopt GloVe [18] to capture the sentence embedding. The image embedding, a 2048-dimension vector, is the output of the final pooling layer. As for the sentence embedding, we adopt the GloVe of 100-dimension version pre-trained on Wikipedia and Gigaword. By feeding both image and sentence embeddings to a following fusion layer and a fully connected layer, the model will output the predicted result. In regard to the design of fusion, we investigate and compare several recent works, such as MCB [12], Mutan [3] and so on. Comparison results among different fusion layers are presented in the experiment part.

Learning Strategy. As for robust learning with noisy labels, following the idea of [13], we exploit the co-teaching method, which filters out wrong labels while

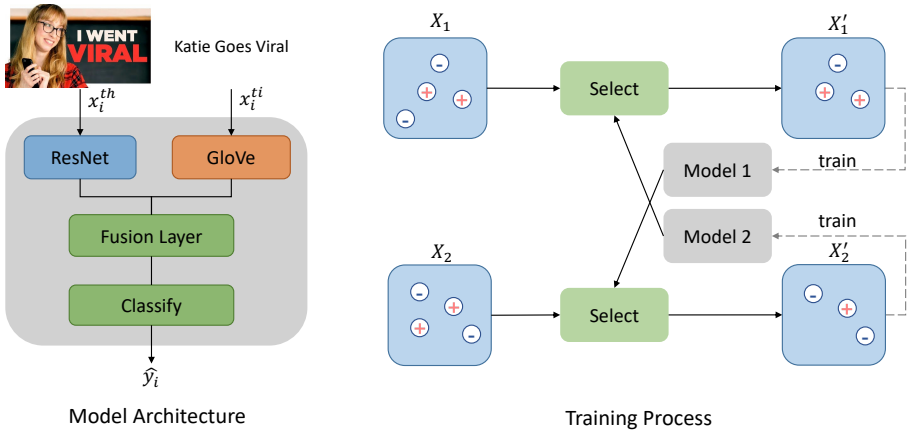

Fig. 4. The architecture of our proposed model and its training process. To tackle noisy labels, we adopt co-teaching to filter out data with wrong labels during training. Note that $X_1, X_2, X_1^{'}, X_2^{'}$ refer to the batch of training samples, and Model 1 and Model 2 share the same model architecture but with different initialized parameters.

training. Concretely, we set up two identical networks to teach each other. In each training batch, each network selects instances with small loss as useful knowledge and teaches these instances to the peer network for further training. The basic assumption behind this strategy is that, on a noisy dataset, deep networks tend to first learn easy and clean patterns in initial epochs. Note that when applying the co-teaching, we oversample the clickbait samples in each batch to make labels of training samples balanced. The reason why we do this is that co-teaching tends to drop positive samples when the number of negative samples is much more. With such configuration, wrongly labeled instances that are out of normal pattern and usually lead to high loss can be removed.

5 Experimental Validation

5.1 Set-Up

After labeling generation, our dataset consists of 787 labeled data and 7,039 weakly labeled data. For evaluation, we select 197 labeled data as the test set while the other 590 data as a part of the training set. Besides, for a fair comparison, we use '5-fold validation' to evaluate each method. Specifically, we conduct 5 experiments for each method and, in each experiment, we select one-fifth from 590 labeled training data as the validation set to pick the best model. The averaged result of 5 experiments is models' final performance.

For our method, we fine-tuned the ResNet while training, and use the average of word embeddings to represent the sentence embedding. For neural network models, we fix batch size as 32 and set the learning rate as 1e-4. We train each

Table 4. Performance comparison of different fusion layer.

Fusion Layer	AUC-ROC	F1 score
ConcatMLP	0.8427	0.6404
Block [4]	0.8452	0.6538
Mutan [3]	**0.8659**	0.6415
BlockTucker [4]	0.8392	0.6329
MFH [27]	0.8603	**0.6585**
MCB [12]	0.8626	0.6170

method for 20 epochs and select the one that performs best in the validation set for evaluation. As for the optimizer, we use Adam with the default hyper-parameters in Pytorch.

Since the label of our test set is not balanced, which is the same case with real data distribution in video-sharing platforms nowadays, we employ F1 score and AUC-ROC as the evaluation metric. The F1 score is the harmonic mean of the precision and recall, which is usually better than accuracy when evaluating with imbalanced labels. AUC-ROC curve is a performance measurement for the classification problems at various threshold settings. ROC is a probability curve and AUC represents the degree of separability. In other words, it represents the capability of the model to distinguish between classes. Note that we also use the AUC-ROC to pick the best model during training.

5.2 Performance Comparison

Fusion Layer Comparison. We first compare several recent works on multimodal fusion using our built model architecture. For this comparison, we only use the labeled data for training models with different fusion Layers. The comparison result is reported in Table 4. Note that ConcatMLP simply concatenates the image embedding and sentence embedding for fusion. For subsequent experiments, we select the Block, Mutan and MFH for further comparison, which perform best among all the fusion layers.

Comparison with Baselines. We then compare our models with several representative and state-of-the-art vision-languages models, including SVM, Logistic Regression, VisualBERT, LXMERT and UNITER. For SVM and logistic Regression, we concatenate the features including the outputs of the pre-trained ResNet-50 and GloVe as their input, which is identical with our model. Note RBF kernel function is used for SVM. As for VisualBERT, LXMERT and UNITER, we use their pre-trained models in the VQA task and fine-tune them to our task.

As shown in Table 5, our methods consistently outperform the baselines. For SVM and Logistic Regression, though they share the same embedding format with our proposed model, their performance is not satisfactory. On one hand, since they are not end-to-end models, they are unable to fine-tune the ResNet

Table 5. Comparison results using different models and with/without generated labels.

	w/o generated labels		w/ generated labels	
Method	AUC-ROC	F1 score	AUC-ROC	F1 score
SVM	0.7149	0.3830	0.7355	0.4000
Logistic Regression	0.7144	0.4912	0.7629	0.5986
VisualBERT [16]	-	-	0.8460	0.6722
LXMERT [25]	-	-	0.8458	0.6640
UNITER [8]	-	-	0.8196	0.6554
Ours + Block [4]	0.8452	0.6538	0.8644	0.6831
Ours + Mutan [3]	**0.8659**	0.6415	**0.8666**	**0.6933**
Ours + MFH [27]	0.8603	**0.6585**	0.8603	0.6884

"-": Does not converge due to a lack of data

during training, which may result in inappropriate image feature representation. In contrast, our end-to-end model does not have such constraint and can fine-tune the ResNet to get a better image feature representation for our task. On the other hand, the fitting and generalization capability of classical machine learning models is not as great as neural networks. As for the current SOTA vision-language models which are based on the transformer, they usually take the object detection results as the input. In this context, they greatly rely on the object detection networks like Faster R-CNN [21], and these networks would not be fine-tuned while training the vision-language models. However, the images used for training objection detection networks are usually different from thumbnails exhibited on video-sharing platforms. In short, there exists a data distribution discrepancy. As a result, the objection results may beyond our expectation and are not ideal for the clickbait thumbnail detection task. That's the possible reason for the limited performance of these BERT-like vision-language models. To improve their performance, an object recognition dataset specific to thumbnails on video websites may be required, which is unavailable currently. Moreover, compared to these transformer-based networks, our model is more light-weight and can adapt to a new domain with much less training data.

Effectiveness of Generated Labels. With weak supervision and majority voter, we generate 7039 labels for unlabeled data with 83.6% accuracy on labeled data. To access the impacts of these generated labels toward models' performance, we make a comparison of models trained with and without generated labels. Note we only have 591 samples for training without generated labels while 7620 samples with generated labels. As reported in Table 5, all the methods benefit from these additional training samples. Experimentally, we found that, without generated labels, transformer-based models are very hard to converge and their training loss barely falls down. This also demonstrates the effectiveness of our generated labels. As for the different improvements in AUC-ROC and F1-score, we think

Table 6. Performance comparison with different forget rate τ.

	F1 socre			AUC-ROC		
Forget Rate	Block	Mutan	MFH	Block	Mutan	MFH
$\tau = 0.00$	0.6831	0.6933	0.6884	0.8644	0.8666	0.8603
$\tau = 0.05$	0.6941	0.6877	0.6759	**0.8714**	0.8663	0.8469
$\tau = 0.15$	0.7102	**0.7122**	**0.7127**	0.8680	**0.8712**	**0.8805**
$\tau = 0.30$	**0.7153**	0.7039	0.7100	0.8672	0.8692	0.8695

that adding generated labels enables models to hold a better decision boundary when the threshold is 0.5, but with a similar ability to distinguish two classes.

5.3 Understanding Co-teaching

Despite the improvement we obtain with generated labels, the performance of models is still limited by the noise in them. In this section, we conduct experiments to verify the effectiveness of co-teaching in combating noisy data, where the choosing of forget rate is critical. Generally, at the initial learning epochs, we can safely update the parameters of the network using all entire noisy data since the network will not memorize the noise in the early stage of training [2]. But as the learning proceeds, the network has to 'forget' some noisy data to prevent fitting them. In other words, we will drop some instances that are considered as noise. And the forget rate means how many instances should be considered as noise and would be dropped in every training batch. To understand how forget rate τ affects the co-teaching, we vary $\tau = \{0, 0.05, 0.15, 0.3\}$ and make a comparison.

Table 6 presents the comparison results of using different forget rate τ. We can observe that all three models benefit from co-teaching, which verifies its effectiveness to tackle noise. Note that $\tau = 0$ means co-teaching is not employed for training. Besides, co-teaching with $\tau = 0.15$ performs better than other forget rate setting. Considering that the accuracy of generated labels is 83.6% in the evaluation, the $\tau = 0.15$ setting helps remove most of the samples with wrong labels at the meanwhile of reserving as many valid training samples as possible, which accounts for the good performance of models in this setting.

5.4 Limitation and Future Work

Our proposed framework CHECKER detects clickbait thumbnails using their visual features in conjunction with their titles without the need to comprehensively process the target videos' contents. This is because CHECKER aims to stimulate the users' experience where ones can detect a clickbait thumbnail even before watching its video. In the future, we hope to explore if utilizing different video comprehension techniques can further improve our model.

6 Conclusion

In this paper, we propose to leverage weak supervision to address the training data shortage in clickbait thumbnail detection. To this end, we first construct a dataset consisted of Youtube videos and invite workers to manually annotate some of them. To make use of unlabeled data, based on characteristics of clickbait thumbnails, we design several high-quality labeling functions as weak supervision sources to generate labels for them. Then, with recent advances in multimodal fusion, we build a multimodal model that takes the thumbnail and title as input to identify clickbait. Furthermore, to deal with noise in generated labels, we adopt co-teaching to filter out samples with wrong labels to train a robust classifier. The experiment results demonstrate the effectiveness of our proposed method.

Acknowledgement

The works of Thai Le and Dongwon Lee were in part supported by NSF awards #1742702, #1820609, #1909702, #1915801, #1934782, and #2114824.

References

1. Agrawal, Amol. "Clickbait detection using deep learning." NGCT, 2016
2. Arpit, Devansh and Jastrzębski, Stanisław and Ballas, Nicolas and Krueger, David and Bengio, Emmanuel and Kanwal, Maxinder S and Maharaj, Tegan and Fischer, Asja and Courville, Aaron and Bengio, Yoshua. "A closer look at memorization in deep networks." ICML, 2017
3. Ben-Younes, Hedi and Cadene, Rémi and Cord, Matthieu and Thome, Nicolas. "Mutan: Multimodal tucker fusion for visual question answering." ICCV, 2017
4. Ben-Younes, Hedi and Cadene, Remi and Thome, Nicolas and Cord, Matthieu. "Block: Bilinear superdiagonal fusion for visual question answering and visual relationship detection." AAAI, 2019
5. Biyani, Prakhar and Tsioutsiouliklis, Kostas and Blackmer, John. "8 amazing secrets for getting more clicks: detecting clickbaits in news streams using article informality." AAAI, 2016
6. Chakraborty, Abhijnan and Paranjape, Bhargavi and Kakarla, Sourya and Ganguly, Niloy. "Stop clickbait: Detecting and preventing clickbaits in online news media." ASONAM, 2016
7. Chen, Mayee F and Fu, Daniel Y and Sala, Frederic and Wu, Sen and Mullapudi, Ravi Teja and Poms, Fait and Fatahalian, Kayvon and Ré, Christopher. "Train and You'll Miss It: Interactive Model Iteration with Weak Supervision and Pre-Trained Embeddings." arXiv:2006.15168, 2020
8. Chen, Yen-Chun and Li, Linjie and Yu, Licheng and El Kholy, Ahmed and Ahmed, Faisal and Gan, Zhe and Cheng, Yu and Liu, Jingjing. "Uniter: Learning universal image-text representations." ECCV, 2020
9. Chen, Yimin and Conroym, Niall J. and Rubin, Victoria L. "Misleading online content: recognizing clickbait as false news." ACM on workshop on multimodal deception detection, 2015

10. Elyashar, Aviad and Bendahan, Jorge and Puzis, Rami. "Detecting Clickbait in Online Social Media: You Won't Believe How We Did It." arXiv:1710.06699, 2017
11. Fu, Daniel and Chen, Mayee and Sala, Frederic and Hooper, Sarah and Fatahalian, Kayvon and Ré, Christopher. "Fast and three-rious: Speeding up weak supervision with triplet methods." ICML, 2020
12. Fukui, Akira and Park, Dong Huk and Yang, Daylen and Rohrbach, Anna and Darrell, Trevor and Rohrbach, Marcus. "Multimodal compact bilinear pooling for visual question answering and visual grounding." arXiv:1606.01847, 2016
13. Han, Bo and Yao, Quanming and Yu, Xingrui and Niu, Gang and Xu, Miao and Hu, Weihua and Tsang, Ivor and Sugiyama, Masashi. "Co-teaching: Robust training of deep neural networks with extremely noisy labels." NIPS, 2018
14. He, Kaiming and Zhang, Xiangyu and Ren, Shaoqing and Sun, Jian. "Deep residual learning for image recognition." CVPR, 2016
15. Le, Thai and Shu, Kai and Molina, Maria D and Lee, Dongwon and Sundar, S Shyam and Liu, Huan. "5 sources of clickbaits you should know! using synthetic clickbaits to improve prediction and distinguish between bot-generated and human-written headlines." ASONAM, 2019
16. Li, Liunian Harold and Yatskar, Mark and Yin, Da and Hsieh, Cho-Jui and Chang, Kai-Wei. "Visualbert: A simple and performant baseline for vision and language." arXiv:1908.03557, 2019
17. Molina, Maria and Sundar, S. Shyam and Roy, Md Main Uddin and Hassan, Naeemul and Le, Thai and Lee, Dongwon. "Does Clickbait Actually Attract More Clicks? Three Clickbait Studies You Must Read." CHI, 2021
18. Pennington, Jeffrey and Socher, Richard and Manning, Christopher D. "Glove: Global vectors for word representation." EMNLP, 2014
19. Qu, Jiani and Hißbach, Anny Marleen and Gollub, Tim and Potthast, Martin. "Towards Crowdsourcing Clickbait Labels for YouTube Videos." HCOMP, 2018
20. Ratner, Alexander and Bach, Stephen H and Ehrenberg, Henry and Fries, Jason and Wu, Sen and Ré, Christopher. "Snorkel: Rapid training data creation with weak supervision." VLDB, 2017
21. Ren, Shaoqing and He, Kaiming and Girshick, Ross and Sun, Jian. "Faster R-CNN: towards real-time object detection with region proposal networks." NIPS, 2015
22. Rony, Md Main Uddin and Hassan, Naeemul and Yousuf, Mohammad. "Diving deep into clickbaits: Who use them to what extents in which topics with what effects?." ASONAM, 2017
23. Shang, Lanyu and Zhang, Daniel Yue and Wang, Michael and Lai, Shuyue and Wang, Dong. "Towards reliable online clickbait video detection: A content-agnostic approach." Knowledge-Based Systems 182, 2019
24. Shu, Kai and Wang, Suhang and Le, Thai and Lee, Dongwon and Liu, Huan. "Deep headline generation for clickbait detection." ICDM, 2018
25. Tan, Hao, and Mohit Bansal. "Lxmert: Learning cross-modality encoder representations from transformers." EMNLP, 2019
26. Xu, Peng and et al. "Clickbait? sensational headline generation with auto-tuned reinforcement learning." EMNLP, 2019
27. Yu, Zhou and Yu, Jun and Xiang, Chenchao and Fan, Jianping and Tao, Dacheng. "Beyond bilinear: Generalized multimodal factorized high-order pooling for visual question answering." IEEE trans. on neural networks and learning systems, pp 5947-5959, 2018
28. Zannettou, Savvas and Chatzis, Sotirios and Papadamou, Kostantinos and Sirivianos, Michael. "The good, the bad and the bait: Detecting and characterizing clickbait on youtube." IEEE Security and Privacy Workshops (SPW), 2018