Class Prototypes based Contrastive Learning for Classifying Multi-Label and Fine-Grained Educational Videos

Rohit Gupta 1 * Anirban Roy 2 Claire Christensen 2 Sujeong Kim 2 Sarah Gerard 2 Madeline Cincebeaux 2 Ajay Divakaran 2 Todd Grindal 2 Mubarak Shah 1 1 Center for Research in Computer Vision, University of Central Florida rohitg@knights.ucf.edu, shah@crcv.ucf.edu 2 SRI International anirban.roy@sri.com

Abstract

The recent growth in the consumption of online media by children during early childhood necessitates data-driven tools enabling educators to filter out appropriate educational content for young learners. This paper presents an approach for detecting educational content in online videos. We focus on two widely used educational content classes: literacy and math. For each class, we choose prominent codes (sub-classes) based on the Common Core Standards. For example, literacy codes include 'letter names', 'letter sounds', and math codes include 'counting', 'sorting'. We pose this as a fine-grained multilabel classification problem as videos can contain multiple types of educational content and the content classes can get visually similar (e.g., 'letter names' vs 'letter sounds'). We propose a novel class prototypes based supervised contrastive learning approach that can handle fine-grained samples associated with multiple labels. We learn a class prototype for each class and a loss function is employed to minimize the distances between a class prototype and the samples from the class. Similarly, distances between a class prototype and the samples from other classes are maximized. As the alignment between visual and audio cues are crucial for effective comprehension, we consider a multimodal transformer network to capture the interaction between visual and audio cues in videos while learning the embedding for videos. For evaluation, we present a dataset, APPROVE, employing educational videos from YouTube labeled with fine-grained education classes by education researchers. APPROVE consists of 193 hours of expert-annotated videos with 19 classes. The proposed approach outperforms strong baselines on APPROVE and other benchmarks such as Youtube-8M, and COIN. The dataset is available at https://nusci.csl.sri.com/project/APPROVE.

*Work partly done during an internship at SRI International.

1. Introduction

With the expansion of internet access and the ubiquitous availability of smart devices, children increasingly spend a significant amount of time watching online videos. A recent nationally representative survey reported that 89% of parents of children aged 11 or younger say their child watches videos on YouTube [4]. Moreover, it is estimated that young children in the age range of two to four years consume 2.5 hours and five to eight years consume 3.0 hours per day on average [46,47]. Childhood is typically a key period for education, especially for learning basic skills such as literacy and math [21, 26]. Unlike generic online videos, watching appropriate educational videos supports healthy child development and learning [7, 23, 24]. Thus, analyzing the content of these videos may help parents, teachers, and media developers increase young children’s exposure to high-quality education videos, which has been shown to produce meaningful learning gains [23]. As the amount of online content produced grows exponentially, automated content understanding methods are essential to facilitate this.

In this work, given a video, our goal is to determine whether the video contains any educational content and characterize the content. Detecting educational content requires identifying multiple distinct types of content in a video while distinguishing between similar content types. The task is challenging as the education codes by Common Core Standards [3, 41] can be similar such as ‘letter names’ and ‘letter sounds’, where the former focuses on the name of the letter and the latter is based on the phonetic sound of the letter. Also, understanding education content requires analyzing both visual and audio cues simultaneously as both signals are to be present to ensure effective learning [3, 41]. This is in contrast to standard video classification benchmarks such as the sports or generic YouTube videos in UCF101 [55] Kinetics400 [54], YouTube-8M [1], where visual cues are often sufficient to detect the different classes. Finally, unlike standard well-known action videos,
education codes are more structured and not accessible to common users. Thus, it requires a carefully curated set of videos and expert annotations to create a dataset that enables a data-driven approach. In this work, we focus on two widely used educational content classes: literacy and math. For each class, we choose prominent codes (sub-classes) based on the Common Core Standards that outline age-appropriate learning standards [3,41]. For example, literacy codes include ‘letter names’, ‘letter sounds’, ‘rhyming’, and math codes include ‘counting’, ‘addition subtraction’, ‘sorting’, ‘analyze shapes’.

We formulate the problem as a multilabel fine-grained video classification task as a video may contain multiple types of content that can be similar. We employ multimodal cues since besides visual cues, audio cues provide important cues to distinguish between similar types of educational content. We propose a class prototypes based supervised contrastive learning approach to address the aforementioned challenges. We learn a prototype embedding for each class. Then a loss function is employed to minimize the distance between a class prototype and the samples associated with the class label. Similarly, the distance between a class prototype and the samples without that class label is maximized. This is unlike the standard supervised contrastive learning setup where inter-class distance is maximized and intra-class distance is minimized by considering classwise positive and negative samples. This approach is shown to be effective for single-label setups [27]. However, it is not straightforward to extend this for the proposed multilabel setup as samples cannot be identified as positive or negative due to the multiple labels. We jointly learn the embedding of the class prototypes and the samples. The embeddings are learned by a multimodal transformer network (MTN) that captures the interaction between visual and audio cues in videos. We employ automatic speech recognition (ASR) to transcribe text from the audio. The MTN consists of video and text encoders that learn modality-specific embedding and a cross-attention mechanism is employed to capture the interaction between them. The MTN is end-to-end learned through the contrastive loss.

Due to the lack of suitable datasets for evaluating fine-grained classification of education videos, we propose a new dataset, called APPROVE, of curated YouTube videos annotated with educational content. We follow Common Core Standards [3,41] to select education content suitable for the kindergarten level. We consider two high-level classes of educational content: literacy and math. For each of these content classes, we select a set of codes. For the literacy class, we select 7 codes and for the math class, we select 11 codes. Each video is associated with multiple labels corresponding to these codes. The videos are annotated by trained education researchers following standard validation protocol [42] to ensure correctness. APPROVE also consists of carefully chosen background videos, i.e., without educational content, that are visually similar to the videos with educational content. APPROVE consists of 193 hours of expert-annotated videos with 19 classes (7 literacy codes, 11 math codes, and a background) where each video has 3 labels on average.

Our contributions can be summarized as follows:

- APPROVE, a fine-grained multi-label dataset of education videos, to promote exploration in this field.
- Class prototypes based contrastive learning framework along with a multi-modal fusion transformer suitable for the problem where videos have multiple fine-grained labels.
- Outperforming relevant baselines on three datasets: APPROVE, YouTube-8M [1] and COIN [56].

2. Related Works

Self-Supervised Contrastive Learning (CL) has been an effective paradigm for visual representation learning. Methods such as SimCLR [8, 9], MoCo (Momentum Contrastive learning) [10, 12, 19], Augmented Multiscale Deep InfoMax (AMDIM) [5], Contrastive Predictive Coding (CPC) [40] and Temporal Contrastive Learning (TCLR) [14] have achieved strong performance on image and video classification benchmarks. The shared property between these CL frameworks is that data augmentation is used to generate positive pairs for CL from a single instance, where other data instances are treated as negatives. Prototypical Contrastive Learning (PCL) [31] extends self-supervised contrastive learning with the idea of clustering data representations during training to generate unsupervised prototypes which represent intra-class variation. We utilize class prototypes instead in the supervised setting, to learn fine-grained distinctions between classes.

Supervised CL methods such as SupCon [27] utilize labels to enhance contrastive learning by forming positive and negative pairs using labels instead of data augmentation. Supervised Contrastive Learning has also been used for other tasks such as image segmentation [58] and classification in the presence of noisy labels [33]. Hierarchical CL [64] extends SupCon to the hierarchical classification case. However, SupCon cannot be extended to the multi-label case in a straightforward manner, as pairs of data samples with multiple labels cannot be clearly classified just into positives and negatives.

Weakly-Supervised Multi-Modal CL: Weakly aligned text-image/video datasets scraped from the web such as Conceptual Captions [51] and WebVid-10M [6] enable learning of multi-modal representations. CLIP [43] applies a cross-modal contrastive loss to train individual text and image encoders. Everything at Once [52] is able to additionally utilize the audio modality and incorporates a pairwise fusion encoder which encodes pairs of modalities, as a re-
result, 6 forward passes of the fusion model are required for 3 modalities. Frozen in Time [6] is able to utilize both image-text and video-text datasets through the use of a Space-Time Transformer Visual Encoder. Visual Conditioned GPT [38] uses a single cross-attention fusion layer to combine pre-trained CLIP text and visual features. Flamingo [2] adds cross-attention layers interleaved with language decoder layers to fuse visual information into text generation. MERTOT [61, 62] and Triple Contrastive Learning [59] combine contrastive learning and generative language modeling to learn aligned text-image representations.

**Supervised Multi-Modal Learning:** Supervised Multi-Modal Learning typically relies on crowd-captioned datasets such as Flickr30k [60] and MS-COCO Captions [11]. Some prior works such as OSCAR [34] and VinVL [63] have utilized pre-trained object detectors and multi-modal transformers to learn image captioning using supervised aligned datasets. BLIP [29] takes a hybrid approach where it bootstraps an image captioner using a labeled dataset and uses it to generate captions for web images. This generated corpus is then filtered and used for learning an aligned representation. ALign BEfore Fuse [30] highlights the importance of aligning text and image tokens before fusing them using a multi-modal transformer.

In this paper, we focus on the fine-grained classification of multilabel educational videos. Due to the lack of suitable datasets, we propose a new dataset, APPROVE, which is described next.

### 3. APPROVE Dataset

We propose a dataset, called APPROVE, of curated YouTube videos annotated with educational content. APPROVE consists of 193 hours of expert-annotated videos with 19 classes (7 literacy codes, 11 math, and background) and each video is associated with approximately 3 labels on average. We follow the Common Core Standards [3, 41] to select education content suitable for kindergarten level. The Common Core Standards outline what students are expected to know and do at various age ranges and grades. This is a widely accepted standard followed by a range of educators. We consider two high-level classes of educational content: literacy and math. For each of these content classes, we select a set of codes. For the literacy class, we select 7 codes including letter names, letter sounds, follow words, sight words, letters in words, sounds in words, and rhyming. For the math class, we select 11 codes including counting, individual number, comparing groups, addition subtraction, measurable attributes, sorting, spatial language, shape identification, building drawing, analyzing and comparing shapes. More details about the standard and the description of the codes are provided in the supplementary material. APPROVE also consists of carefully chosen background videos, i.e., without educational content, that are visually similar to the videos with educational content. We present frames corresponding to these classes in Fig. 1.

![Sample video frames from the APPROVE dataset](image)

**Figure 1.** Sample video frames from the APPROVE dataset. Videos belong to the (a) literacy classes, (b) math classes, and (c) background. Background videos do not contain educational content but share visual similarities with educational videos. The videos are labeled with fine-grained sub-categories, e.g., letter names vs letter sounds.

![Frequency of the classes in APPROVE](image)

**Figure 2.** Frequency of the classes in APPROVE. Math codes are in Orange and literacy codes in Blue.

To ensure the quality and correctness of the annotations, we consider educational researchers to annotate the videos and follow a standard validation protocol [42]. Each annotator is trained by an expert and annotations on a selected set are examined before engaging the annotator for the final annotation. Annotators start once they reach more than 90% agreement with the expert. Further, we estimate inter-
annotator consistency to filter out anomalies. Details about the validation process are provided in the supplementary material. It takes a month to train an education researcher to match expert-level coding accuracy. On average, it takes the trained annotators 1 min to annotate 1 min of video.

The videos are curated from YouTube and are annotated by the trained annotators to determine educational content in them. Each video can have multiple class labels that are quite similar making the task a multi-label and fine-grained classification problem. For example ‘letter names’ and ‘letter sounds’ where visual letters are shown in both but in ‘letter sounds’, the phonetic sound on the letter is emphasized (Fig. 1 (a)). Similarly, in both ‘build and draw shapes’ and ‘analyzing and comparing shapes’, multiple shapes can appear but the latter focuses on comparing multiple shapes by shape and size (Fig. 1 (b)). Class-wise stats are presented in Fig. 2. Note that the task is different from common video classification setups where either multi-label or fine-grained aspects are dealt separately. Single-label datasets such as HMDB51 [28], UCF101 [55], Kinetics700 [54] and multi-label ones such as Charades [53] are widely used benchmarks for this problem. YouTube-Birds and YouTube-Cars [65] are analogous datasets for object recognition from videos. Multi-Sports [35] and FineGym [50] label fine-grained action classes for sports. HVU [16] also adds scenes and attributes annotations along with action and objects. However, action, object and scene recognition are not enough for fine-grained video understanding. For instance, videos from a given education provider might share similar objects (person, chalkboard, etc.) and actions (writing on chalkboard) while covering different topics (counting, shape recognition etc.) in each video.

4. Proposed Approach

In this section, we first describe the proposed class prototypes based contrastive learning framework suitable for videos containing multiple educational codes. Then we present the approach to learning the class prototypes and finally describe the multimodal transformer network that learns features by fusing visual and text cues from videos.

### 4.1. Class prototypes based contrastive learning

In a contrastive learning framework, feature representations are typically learned by simultaneously minimizing the distance between positive samples and maximizing the distance between negative samples (See Figure 4.(a)). The positive and negative samples are determined with respect to an anchor sample usually based on the class labels.

For example, supervised contrastive learning (SupCon) [27] learns a representation to minimize the intra-class distances and maximize inter-class distances. We denote $x_i$ and $y_i$ as the $ith$ sample and its label, respectively. Let’s define $z_i$ as the representation of the $ith$ sample in a batch $A$, and $sim(z_i, z_j) = \frac{z_i \cdot z_j}{||z_i||^2}$ the cosine similarity, then the SupCon loss [27] is defined as:

$$L_{SupCon} = \sum_{i \in A} \sum_{p \in P(i)} \log \frac{\exp(sim(z_i, z_p)/\tau)}{\sum_{a \in A \setminus i} \exp(sim(z_i, z_a)/\tau)}$$

where $P(i)$ is the set of positive samples, i.e., with the same label as $z_i$, in the batch excluding $i$ and $a \in A \setminus i$ is the index of all samples in the batch excluding the $ith$ sample. $\tau$ is a scalar temperature parameter used for scaling similarity values. The positive pairs are grouped into the numerator while determining positive and negative samples. SupCon is known to be effective for classifying samples with a single label. However, it is not straightforward to extend this for the multilabel setup as beyond positive samples, where all labels are the same, and negative samples, where none of the labels is the same, there can be a third scenario where labels are partially overlapping. Though SupCon has been extended to hierarchical classification [64], it cannot be directly extended to the true multi-label case.

To address this issue, we learn class prototypes as the representative for each class and consider these as anchors while determining positive and negative samples. Specifically, for a specific class prototype, a representation is learned to minimize distances between the prototype and samples with this class label and maximize the distances between the prototype and samples without this class label. We compare the proposed approach with the standard single-label contrastive learning in figure 4. We iteratively update the class prototypes while learning the feature rep-

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Size (in hr)</th>
<th>Multi-Label</th>
<th>Fine Grained</th>
<th>Type</th>
<th>Annotators</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action Recognition</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HMBD</td>
<td>5</td>
<td>✗</td>
<td>✗</td>
<td>V</td>
<td>Authors</td>
</tr>
<tr>
<td>UCF</td>
<td>27</td>
<td>✗</td>
<td>✗</td>
<td>V</td>
<td>Authors</td>
</tr>
<tr>
<td>Kinetics</td>
<td>800</td>
<td>✗</td>
<td>✗</td>
<td>V+A</td>
<td>Crowd</td>
</tr>
<tr>
<td>Video Classification</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COIN</td>
<td>476</td>
<td>✗</td>
<td>✗</td>
<td>V+A</td>
<td>Crowd</td>
</tr>
<tr>
<td>YT-8M</td>
<td>-</td>
<td>✓</td>
<td>✗</td>
<td>F</td>
<td>Machine</td>
</tr>
<tr>
<td>APPROVE</td>
<td>193</td>
<td>✓</td>
<td>✓</td>
<td>V+T+A</td>
<td>Experts</td>
</tr>
</tbody>
</table>

Table 1. APPROVE dataset compared with selected prior datasets. V→Video Frames, A→Audio, T→Text, F→Features only.
Contrastive learning operates on the feature space by bringing the representations of similar samples close and pushing distinct samples apart. Prior work in a) Supervised Contrastive Learning [27] trains the network by treating instances from the same class as positive pairs and instances from different classes as negative pairs. This approach doesn’t generalize to multi-label classification tasks, as some instance pairs have partially overlapping labels. We propose the use of class prototypes to enable b) Multi-Label Prototypes Contrastive Learning. Each sample and the class prototypes corresponding to the labels associated with the sample are treated as positive pairs. Similarly, negative pairs are determined based on the missing class labels. Prototypes are represented by stars (⋆) and inputs as circles (○) colored with all their relevant labels. We discuss strategies for initializing and learning the label prototypes in Sec. 4.2.

Figure 4. Contrastive learning operates on the feature space by bringing the representations of similar samples close and pushing distinct samples apart. Prior work in a) Supervised Contrastive Learning [27] trains the network by treating instances from the same class as positive pairs and instances from different classes as negative pairs. This approach doesn’t generalize to multi-label classification tasks, as some instance pairs have partially overlapping labels. We propose the use of class prototypes to enable b) Multi-Label Prototypes Contrastive Learning. Each sample and the class prototypes corresponding to the labels associated with the sample are treated as positive pairs. Similarly, negative pairs are determined based on the missing class labels. Prototypes are represented by stars (⋆) and inputs as circles (○) colored with all their relevant labels. We discuss strategies for initializing and learning the label prototypes in Sec. 4.2.

Figure 4. Contrastive learning operates on the feature space by bringing the representations of similar samples close and pushing distinct samples apart. Prior work in a) Supervised Contrastive Learning [27] trains the network by treating instances from the same class as positive pairs and instances from different classes as negative pairs. This approach doesn’t generalize to multi-label classification tasks, as some instance pairs have partially overlapping labels. We propose the use of class prototypes to enable b) Multi-Label Prototypes Contrastive Learning. Each sample and the class prototypes corresponding to the labels associated with the sample are treated as positive pairs. Similarly, negative pairs are determined based on the missing class labels. Prototypes are represented by stars (⋆) and inputs as circles (○) colored with all their relevant labels. We discuss strategies for initializing and learning the label prototypes in Sec. 4.2.

Contrastive Learning

L_{\text{mlc}}(x) = -\frac{1}{|P_{\text{ml}}(x)|} \sum_{c^+_k \in P_{\text{ml}}(x)} \left[ \log \sum_{c^-_j \in C \setminus P_{\text{ml}}(x)} \frac{\exp(sim(z, c^-_j)/\tau)}{\exp(sim(z, c^-_j)/\tau)} \right]

(2)

Here, minimizing the loss of the positive class prototype and instance pairs in the numerator minimizes the distance between the representation z and the class prototypes corresponding to the sample, and vice versa for negative classes. We also utilize negative sampling to account for the class imbalance between positives and negatives.

4.2. Learning class prototypes

We aim to learn class-specific prototypes such that the multilabel samples can be thought of as the combinations of the class prototypes selected based on the associated labels. Let’s assume Z_t is an N x d matrix of d-dimensional representations, i.e., z_i^s, of N samples and L \in \{0, 1\}^{N \times K} is corresponding labels matrix with K classes. Let’s denote CP_t is a matrix of size K x d of K class prototypes at a training iteration t. Then, Z_t = L \times CP_t + \varepsilon, where \varepsilon is the residual noise term. Assuming a Gaussian noise that is unbiased and uncorrelated with the labels L, the class prototypes can be approximated as CP_t^*_t \approx (L^T L)^{-1} L^T Z_t, where operation (L^T L) results in a square matrix amenable to inversion. Note that for single labels, this implies averaging the features of the instances belonging to a given class as the prototype for that class. In a multi-label setup, additionally, the co-occurrence between the labels is considered. Finally, the class prototypes are updated with learning iterations as:

CP_{t+1} = \beta \cdot CP_t + (1 - \beta) \cdot CP_t^*,

(3)

where \beta is the decay parameter for the exponential moving averaging. The moving averaging avoids collapsing prototypes across the training iterations [32].

4.3. Multi-Modal Fusion Transformer

Given a sample video x, we consider multimodal cues such as visual and audio cues to learn its representation z. Considering multimodal cues is crucial for content recognition, specifically for education videos as effective comprehension requires attending to both visual demonstration and audio explaining the educational content [3, 41]. To capture audio cues, we consider the audio track of the video and extract speech by removing the background such as tunes or instruments [22]. Then we employ the automatic speech recognition (ASR) technique to transcribe text from the speech [44]. We notice that separating speech from the background is important for an accurate ASR transcription. Given the video frames and text transcription, we propose a multimodal transformer network (MTN) to fuse these cues using cross-modal attention.

Our MTN (Figure 5) has three components: image encoder, text encoder, and fusion encoder to learn visual (z^v), text (z^t), and fusion (z^f) representations, respectively. The sample representation is comprised of these three representations z = {z^v, z^t, z^f}. The image encoder is implemented by a vision transformer (ViT) [17] that learns frame embeddings from the video frames along with a special CLS for each frame [15]. We pool the CLS tokens across
4.4. Inference using prototypes

Existing approaches in contrastive learning [8, 27] usually discard the MLP layer after the contrastive training, and a linear classifier is trained on the frozen backbone network. We rely on the class prototypes to carry out inference by utilizing the cosine distance between the learned prototypes and test features. Given our prototype loss-based training, the estimated probability of a given class should be proportional to the normalized temperature scaled cosine distance (Equation 4). In practice, we normalize the cosine distance such that $-1$ and $1$ correspond to a confidence of $0$ & $1$ respectively. Thus the prediction is made following:

$$\hat{p}(k|x) \propto \exp(sim(z, cp_k)/\tau),$$

(4)

where $z$ is the multimodal representation of the sample $x$.

4.5. Implementation Details

**Training strategy for the multimodal fusion:** We follow a two-stage training process: during the initial unimodal training phase, we utilize fixed prototypes in each modality to align the representations. Then in the second stage, we train the unimodal encoders and the multi-modal fusion encoder end-to-end. The cross-modal alignment learned during the first stage improves the learning of the multi-modal representation. The multi-modal learning phase consists of alternating optimization steps of training the network using our contrastive loss per Sec. 4.1 and refining the class prototypes as described in Sec. 4.2.

**Image encoder:** For video frames, Random Resized Crop and RandAugment [13] augmentations are used from torchvision. We use ImageNet pretrained vision encoders ResNet50 [20], ViT-B/32 (224×224 resolution) and ViT-B/16 (384×384 resolution) [18].

**Text encoder:** We generate text from the videos using Whisper [44], an open source ASR model. For data augmentation, we generate four versions of the ASR text by back-translation using the Helsinki-NLP/opus-mt-\{en-de, en-nl, en-fr\} models through the nlpaug library [39]. Synonym replacement, text span removal and random word swapping augmentations are also used for the text data. We use DistilBERT-Base-uncased [49], and t5-small [45] from HuggingFace transformers library as the text encoder.

**Optimizer:** We employ AdamW optimizer [37] for training with a learning rate of 0.0005. Weight decay of 1e-6 is utilized only on the MLP head during contrastive training and the classifier during BCE/Focal/Asym. loss. Since pre-trained vision and text backbones are used, the backbone learning rate is set to 1/10th of the learning rate for the head. Exponential Moving Averaging every 10 steps with a decay of 0.999 was used for the model parameters.

5. Experiments

**Datasets.** In addition to the proposed APPROVE dataset, we evaluate our approach on a subset of Youtube-8M [1] and COIN [56] datasets. YT-8M consists of a diverse set of YouTube videos with video and audio modalities. We consider a subset of YT-8M dataset with 46K videos and 165 classes. COIN consists of instructional videos covering a wide variety of domains and spanning over 180 classes.

**Baselines.** We compare the efficacy of our multilabel classification framework against the following baselines:
**Binary cross-entropy:** In this baseline, loss for multiple labels is computed by combining the binary cross-entropy losses for individual classes.

**Focal loss [36]:** This considers a modified binary cross-entropy to assign a higher weight to hard samples by adjusting a focusing parameter $\gamma$. Negative samples can also be down-weighted by using a weight $\alpha$. The focal loss for a positive label is given as $\mathcal{L}_{focal}(p) = -\alpha(1-p)^\gamma \log(p)$. We set $\gamma = 2$ and $\alpha = 0.2$ in our experiments.

**Asymmetric loss [48]:** This builds upon the focal loss by utilizing different focusing parameters $\gamma_+$ and $\gamma_-$ for positive and negative labels. It also ignores the negative samples with a prediction probability lower than a margin $m$. Asymmetric Loss for prediction $p$ corresponding to label $y$ is given as: $L_{asym}(y, p) = -y L_+ - (1-y) L_-$, where $L_+ = (1-p)^{\gamma_+} \cdot \log(p)$ and $L_- = (\max(p-m, 0))^{\gamma_-} \cdot \log(1-\max(p-m, 0))$. We follow the 5-step procedure recommended by the original authors to train this baseline. We experimentally set $\{\gamma_- = 2, \gamma_+ = 1, m = 0.1\}$ corresponding to the best performance on APPROVE.

**Metrics.** In order to develop a reliable education content detection framework, achieving high precision is crucial. Thus we consider Recall @ 80% Precision (R@80) as the primary metric. We also consider the standard area under the precision-recall curve (AUPR) that is more suitable for the multilabel setup. This estimates whether the ground truth classes are predicted with higher scores than the rest:

$$LRAP = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{|Y_i|} \sum_{\lambda \in Y_i} |Y_i^\prime : rnk_i(\lambda') \leq rnk_i(\lambda)|,$$

where $rnk_i(\lambda)$ is the predicted rank of class $\lambda$ for sample $i$. LRAP is a ranking metric and independent of a threshold.

**Results on APPROVE.** We compare the proposed approach with the baselines in Tab. 2. Our approach outperforms the strongest baselines by 3.1% and 2.3% with respect to R@80 and AUPR, respectively. We also present results for separate models trained on Math and Literacy subsets of APPROVE respectively. Results on the Math subset are higher compared to the Literacy subset, which indicates the literacy classes are harder to distinguish mostly due to the high inter-class similarity. The top three hardest classes are follow_words, letters_in_words, and sounds_in_words and these are from the literacy set.

**Results on YT-46K and COIN.** Beyond the APPROVE dataset, we also test our approach on two public datasets: YT-46K and COIN. Here we provide the results for the multi-modal models and the results for the single-modality models are in Section D of the Supplementary Material.

Results on YT-46K are provided in Table 3. As YT-8M was primarily collected with the intention of visual classification, the additional use of text data leads to a smaller improvement compared to APPROVE. Results on COIN are provided in Table 4. As each video from COIN is mapped to a single task, we consider the Top-1 accuracy as the metric. On COIN we compare our approach with SupCon [27] which is effective for single labels. Note that our approach outperforms SupCon and this justifies the effectiveness of the class prototypes based training in a generic contrastive learning framework.

### 5.1. Ablation Studies

We perform the following ablation studies to quantify the impact of our approach for learning class prototypes, the multimodal fusion module, and the choice of visual and text encoding frameworks.

<table>
<thead>
<tr>
<th>Subset</th>
<th>Modality</th>
<th>Method</th>
<th>AUPR</th>
<th>LRAP</th>
<th>R@80</th>
</tr>
</thead>
<tbody>
<tr>
<td>V→T</td>
<td>BCE</td>
<td>45.5</td>
<td>54.3</td>
<td>6.9</td>
<td></td>
</tr>
<tr>
<td>V→T</td>
<td>Focal</td>
<td>45.9</td>
<td>56.6</td>
<td>15.0</td>
<td></td>
</tr>
<tr>
<td>V→T</td>
<td>Ours</td>
<td>46.7</td>
<td>57.9</td>
<td>19.6</td>
<td></td>
</tr>
</tbody>
</table>

| All    | T        | BCE    | 79.8 | 85.1 | 63.3 |
| All    | Focal    | 79.9   | 85.7 | 72.8 |
| All    | Ours     | 82.5   | 87.4 | 75.4 |

| MTH    | V+T      | BCE    | 84.3 | 88.4 | 76.3 |
| MTH    | Focal    | 86.1   | 89.1 | 82.2 |
| MTH    | Asym.    | 86.0   | 89.2 | 82.4 |
| MTH    | Ours     | 88.4   | 90.7 | 85.5 |

| LIT    | V+T      | BCE    | 72.1 | 82.9 | 50.7 |
| LIT    | Focal    | 72.7   | 83.5 | 59.9 |
| LIT    | Ours     | 73.6   | 84.7 | 54.7 |

Table 2. Results on APPROVE dataset. All metrics in %.

<table>
<thead>
<tr>
<th>Modality</th>
<th>Method</th>
<th>AUPR</th>
<th>LRAP</th>
<th>R@80</th>
</tr>
</thead>
<tbody>
<tr>
<td>V+T</td>
<td>BCE</td>
<td>64.6</td>
<td>70.2</td>
<td>42.3</td>
</tr>
<tr>
<td>V+T</td>
<td>Focal  [36]</td>
<td>69.7</td>
<td>72.7</td>
<td>44.6</td>
</tr>
<tr>
<td>V+T</td>
<td>Ours</td>
<td>70.9</td>
<td>74.9</td>
<td>49.1</td>
</tr>
</tbody>
</table>

Table 3. Results on YT-46K. V→Video Frames and T→Text.

<table>
<thead>
<tr>
<th>Modality</th>
<th>Method</th>
<th>Top-1 Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>V+T</td>
<td>CE</td>
<td>53.7</td>
</tr>
<tr>
<td>V+T</td>
<td>BCE</td>
<td>54.9</td>
</tr>
<tr>
<td>V+T</td>
<td>Focal  [36]</td>
<td>56.1</td>
</tr>
<tr>
<td>V+T</td>
<td>SupCon [27]</td>
<td>54.7</td>
</tr>
<tr>
<td>V+T</td>
<td>Ours</td>
<td>57.5</td>
</tr>
</tbody>
</table>

Table 4. Results on COIN. V→Video Frames and T→Text.
Learning class prototypes: We compare the two strategies where after initializing the class prototypes, we 1) keep them fixed and learn only the multimodal embedding of the samples, and 2) class prototypes and sample embedding are learned iteratively. The initialization can be done either randomly, or with orthogonal constraints. We note that orthogonal initialization performs best in our experiments and iterative adjusting the class prototypes achieves better performance as shown in Table 5 (a). We also consider hierarchical prototypes, for APPROVE, using a 2-level hierarchy where the first level consists of 18 classes, and the second level is the 3 super-classes: math, literacy, and background. The 180 task categories of COIN are organized into 12 domains in the taxonomy provided with the dataset. This hierarchy imposes an additional constraint on learning the embeddings during training.

Fusion Encoder: This evaluates the effect of the number of layers in the fusion encoder on the final performance. As expected, the performance improves with more layers and saturates around 4 layers (Table 5 (b)).

Vision Encoder: This evaluates the effect of the image encoder on the final performance. We consider ResNet [20] and ViT variants [17]. We notice that the ViT-B/16-384 encoder works well for the larger COIN dataset, whereas the ViT-B/32-224 encoder works best for APPROVE (Table 5 (c)).

Text Encoder: This evaluates the effect of the text encoder on the final performance. We test DistilBERT and T5 backbones for the text encoder. BERT is trained to predict masked spans of text. T5’s unsupervised objective is similar, however, it trains on predicting the entire sequence instead of just the masked spans. GPT2 takes an auto-regressive approach to language modeling (Table 5 (d)).

5.2. Robustness analysis

We evaluate the robustness of our approach as follows:

Noisy modality: YouTube videos may have noisy modalities where some of the video frames are missing or ASR transcription is noisy. We show that our approach is robust against the cases where a percentage of video frames or text words are missing as shown in Tab 6(a).

Run-to-Run variance: The low variance across runs (Tab 6(b)) indicates that our approach is not sensitive to random initialization of the class prototypes.

Initializing the encoders: We consider the ImageNet pre-training for the image encoder, while English Wikipedia + Toronto Book Corpus is used to pre-train the text encoder. We provide results where the backbones are initialized with CLIP [43], which provides a more aligned vision-text representation. As expected, the results are better with the CLIP initialization (Tab 6(c)). The improvements are more significant on COIN than APPROVE as CLIP models may not be exposed to educational videos.

6. Conclusion

We have proposed an approach for detecting educational content in online videos. The problem is formulated as a fine-grained multilabel video classification task and we have considered class-prototypes based contrastive learning to address this. We have employed a multimodal transformer network to fuse visual and audio cues. This is crucial for comprehending educational content as both visual and audio cues are to be aligned to ensure effective comprehension. Our approach is shown to be effective in distinguishing fine-grained educational content with high inter-class similarity. We have introduced APPROVE - a dataset with 193 hours of expert-annotated educational videos. Beyond APPROVE, we have evaluated our approach on COIN and YouTube-8M datasets where our approach outperforms the competitive baselines.

Acknowledgements

This work is supported by NSF grant # 2139219 and SRI R&D award. Rohit Gupta is supported by ARO grant W911NF-19-1-0356. Views and conclusions contained herein are those of the authors and should not be interpreted as representing the official policies or endorsements, either expressed or implied, of NSF, ARO, IARPA, DOI/IBC, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.

Table 5. Ablation studies. R@80% Precision for APPROVE and Top-1 Accuracy for COIN. Default setup is underlined.

<table>
<thead>
<tr>
<th>Variant</th>
<th>APPROVE</th>
<th>COIN</th>
<th>Layers</th>
<th>APPROVE</th>
<th>COIN</th>
<th>Vision Model</th>
<th>APPROVE</th>
<th>COIN</th>
<th>Text Model</th>
<th>APPROVE</th>
<th>COIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>84.1</td>
<td>56.6</td>
<td>1</td>
<td>84.9</td>
<td>57.1</td>
<td>R50</td>
<td>84.8</td>
<td>55.2</td>
<td>DistilBERT-B</td>
<td>85.5</td>
<td>57.5</td>
</tr>
<tr>
<td>Orthogonal</td>
<td>84.8</td>
<td>57.0</td>
<td>2</td>
<td>85.5</td>
<td>57.5</td>
<td>ViT-B/32</td>
<td>85.5</td>
<td>57.5</td>
<td>t5-S</td>
<td>87.3</td>
<td>57.9</td>
</tr>
<tr>
<td>Learned</td>
<td>85.5</td>
<td>57.5</td>
<td>4</td>
<td>85.3</td>
<td>57.6</td>
<td>ViT-B/16</td>
<td>83.8</td>
<td>57.8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hierarchical</td>
<td>86.0</td>
<td>57.8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 6. Robustness analysis. Our method is robust to partially missing modality and has similar run-to-run variance as baselines.

<table>
<thead>
<tr>
<th>% missing</th>
<th>APPROVE</th>
<th>COIN</th>
<th>Method</th>
<th>APPROVE</th>
<th>COIN</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>0%</td>
<td>83.5</td>
<td>57.5</td>
<td>BCE</td>
<td>76.3</td>
<td>57.5</td>
<td></td>
</tr>
<tr>
<td>10% V</td>
<td>80.1</td>
<td>53.3</td>
<td>Focal</td>
<td>82.2</td>
<td>56.1</td>
<td>Wiki-en+TBC</td>
</tr>
<tr>
<td>10% T</td>
<td>75.8</td>
<td>57.2</td>
<td>Ours</td>
<td>85.5</td>
<td>57.5</td>
<td>CLIP</td>
</tr>
<tr>
<td>30% T</td>
<td>68.9</td>
<td>42.8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 6. Robustness analysis. Our method is robust to partially missing modality and has similar run-to-run variance as baselines.
Supplementary Material

This supplementary material is organized into four sections. In Section A we provide additional details about our APPROVE dataset. Section B presents detailed classwise result for our models, demonstrating the impact of multi-modal learning. In Section C we analyze the feature space of our trained model to demonstrate that it picks up semantic similarities between the classes. Finally, we provide implementation details in Section D to assist reproducibility. Note that this document contains 12 pages and some are partially blank to clearly separate different parts of the material.

A. APPROVE Dataset

The key property of APPROVE that sets it apart from prior datasets is its fine-grained and multi-label nature. We provide some visualizations here to build on the main paper and illustrate these properties.

A.1. Fine-Grained

Additional samples from the Literacy (in Figure 6) and Math (in Figure 7) splits of APPROVE are visualized here. These examples are randomly picked, and they highlight the fine-grained nature of the dataset. Many pairs or groups of classes in APPROVE have very high visual similarity, e.g. Shape ID and Building Drawing Shapes, Sounds in Words and Rhyming, etc. Also note that background, math and literacy are distinct and do not share overlapping labels, which illustrates the hierarchical structure of APPROVE.

A.2. Multi-Label

APPROVE is a densely multi-labelled dataset. The multi-label co-occurrence matrix for APPROVE is visualized in Figure 8. Each cell of the matrix, $L[i, j]$, equals the fraction of videos with class $i$ which also contain class $j$. Note that the matrix is not symmetric as two classes might have a one-sided relationship. e.g. presence of written numerals suggests comparing groups is highly likely to be present, however the inverse does not hold. since many videos teach how to compare groups without using written numberals, e.g. comparing groups of objects by some non-numeric property such as a shape or color.

A.3. Class descriptions

Detailed description of the education codes corresponding to each class in APPROVE and their annotation criteria are presented in Table 7. These fine-grained classes correspond to age-appropriate curriculum topic recommendations prescribed by the common core education standards. These detailed descriptions along with a large batch of examples was provided to all data annotators to ensure high quality labeling.

A.4. Annotation evaluation

To ensure the quality and correctness of the annotations, we consider educational researchers to annotate the videos and follow a standard validation protocol [42]. We consider two expert annotators and a few education researchers for annotating the videos. Experts train annotators to identify the indicators of educational content in videos. After training, education researchers are evaluated on a validation set of 50 videos that are already annotated by two experts. Annotators are allowed to start the final annotation process once they achieve more than 90% agreements with the expert annotations. We observe a 95% agreement is reached after four weeks of training. We also consider inter-annotator consistency for the final annotations. Videos that are not consistently labeled by all the annotators are ignored.

A.5. Education codes

APPROVE consists of 193 hours of videos with 19 classes including 7 literacy codes, 11 math, and background. We follow the Common Core Standards [3, 41] to select education content suitable for the kids at kindergarten level. Descriptions of these codes are provided in Tab. 7. Some sample frames for these codes are presented in Fig. 6 and 7.
a. sounds in words
b. rhyming
c. sight words
d. letter sounds
e. follow words

Figure 6. Sample frames from five Literacy classes in APPROVE. The classes share visual similarity, which makes classification a challenging fine-grained learning task.
Figure 7. Sample Frames from five Math classes in APPROVE.
Figure 8. Ground-Truth Multi-Label co-occurrence matrix for APPROVE videos. The three high level groups of categories: Background, Math and Literacy can be seen, highlighting the hierarchical structure of the dataset.
Table 7. Description of the education codes used in APPROVE. These codes correspond to age-appropriate curriculum recommendations prescribed by the common core education standards.

<table>
<thead>
<tr>
<th>Code names</th>
<th>High-level class</th>
<th>Description of the code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Counting</td>
<td>Math</td>
<td>• More than one number in the standard sequence. Starting point can be any whole number.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• This includes counting parts of a shape, such as counting sides or vertices.</td>
</tr>
<tr>
<td>Written numerals</td>
<td>Math</td>
<td>• A written numeral, either on its own or as part of a count sequence, with corresponding visual or audio.</td>
</tr>
<tr>
<td>Quantity of objects</td>
<td>Math</td>
<td>• Emphasizing that the last number in a count sequence represents the total number of objects.</td>
</tr>
<tr>
<td>Addition or subtraction</td>
<td>Math</td>
<td>• At least two numbers and the number that results when they are added or subtracted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Includes adding by counting on. For example, “We have three, let’s add two. Three, four, five. Three and two make five.”</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Includes decomposing sets of objects into two or more sets.</td>
</tr>
<tr>
<td>Measurable attributes</td>
<td>Math</td>
<td>• Describing one object or comparing multiple objects based on at least one measurable attribute, such as length or volume.</td>
</tr>
<tr>
<td>Comparing groups</td>
<td>Math</td>
<td>• Comparing two or more groups of objects.</td>
</tr>
<tr>
<td>Sorting</td>
<td>Math</td>
<td>• Objects being sorted into categories, such as but not limited to color, shape, object type, purpose, pattern, or species.</td>
</tr>
<tr>
<td>Spatial language</td>
<td>Math</td>
<td>• Words and visuals to describe position or movement.</td>
</tr>
<tr>
<td>Shape identification</td>
<td>Math</td>
<td>• Naming and displaying a shape.</td>
</tr>
<tr>
<td>Building or drawing shapes</td>
<td>Math</td>
<td>• Showing how a geometric shape is drawn or built.</td>
</tr>
<tr>
<td>Analyzing or comparing shapes</td>
<td>Math</td>
<td>• Describing one or more shapes in terms of their attributes.</td>
</tr>
<tr>
<td>Letter names</td>
<td>Literacy</td>
<td>• Any spoken or sung letter name.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Does not need to say whether the letter is upper- or lowercase.</td>
</tr>
<tr>
<td>Letter sounds</td>
<td>Literacy</td>
<td>• Any spoken or sung letter sound. Must be distinct from a spoken word.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Can include letter names if they are also letter sounds (e.g., long forms of vowels).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Does not need to say whether the letter is upper- or lowercase.</td>
</tr>
<tr>
<td>Letters in words</td>
<td>Literacy</td>
<td>• A letter within a word is visually highlighted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the video names multiple letters within a word, to meet this criterion it must highlight each letter individually as it is named.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the video separately names the letter and then displays it in a word, the letter must be visually highlighted within the word.</td>
</tr>
<tr>
<td>Sight words</td>
<td>Literacy</td>
<td>• Only words on the sight words list count for this code. As long as a video includes at least one word on the sight words list, this indicator is present.</td>
</tr>
</tbody>
</table>

Continued on next page
<table>
<thead>
<tr>
<th>Code names</th>
<th>High-level class</th>
<th>Description of the code</th>
</tr>
</thead>
</table>
| Sounds in words | Literacy         | • The sound may occur anywhere within a word, including the beginning sound.  
|                 |                  | • The sound must not be the full word.  
|                 |                  | • Choose response option, sound of individual letter if the video includes audio of the sound a single letter makes within a word.  
|                 |                  | • Choose response option, sound of multiple letters together if the video includes audio of the sound of two or more letters together within a word, excluding the full word. For example, “at” in “rat.”  
|                 |                  | • Can include separately making the sound of a letter on its own and displaying it in a word, so long as both occur within about 2 seconds.  
|                 |                  | • Still counts even if other words are used between the full word and the sound. For example, “The words cat and rat both have the ‘t’ sound at the end.” |

| Follow words    | Literacy         | • Must show a passage containing multiple words. Only one word on screen at a time would not count.  
|                 |                  | • Words must be highlighted left to right, top to bottom, and/or page to page. Highlighting can include one word in a passage appearing at a time.  
|                 |                  | • It’s okay if words aren’t highlighted exactly as they are spoken (e.g., highlighting an entire line of text in a paragraph at a time, highlighting words at a constant pace that doesn’t totally line up with audio), so long as the highlighting generally moves left to right or top to bottom as the words are spoken.  
|                 |                  | • Includes sing-along style videos that highlight words as they are sung. |

| Rhyming         | Literacy         | • Within 60 seconds of the word ”rhyme” or ”rhyming,” audio of at least 2 rhyming words.  
|                 |                  | • “Rhyme” may occur before or after the rhyming words.  
|                 |                  | • Rhyming words do not need to be spoken one after the other (e.g., “cheese, please”); they could have words between them, such as a poem or song (e.g., the cat jumped over the hat). |

End of Table 1
B. Classwise Results

We demonstrated strong overall results in the main paper. In particular we found that using multi-modal input data resulted in strong results. Here, we provide class-wise recall and F1 scores in Figure 9 and Figure 10 respectively. These show that our improvements occur across a wide variety of video classes. In order to compute Recall and F1 score, we pick the classification threshold to achieve 80% overall precision to satisfy the requirements of the sensitive education application scenarios (as discussed in the main paper). The threshold found are 0.91 for Video only model, 0.56 for the Text only model and 0.51 for the Video+Text model. As can be noticed in Figure 9 Text only model generally outperforms the Video only model, but the Text+Video model outperforms the Text only model for most classes. Classes which focus on skills requiring connecting language to vision such as Sight Words, Written Numerals and Sorting benefit the most from the use of multi-modal data for classification.

In Figure 11 we provide a scatter plot of class-wise recall for the text only model recall vs the video only model. The recall is weakly correlated across the two modalities ($R^2 = 0.122$), which explains the significant gains due to combining the two modalities.

![Figure 9. Classwise Recall at 80% overall Precision. Most classes benefit from access to multi-modal input data.](image-url)
Figure 10. Classwise F1 Scores at 80% overall Precision.

Figure 11. Comparing classwise recall between video and text models.
Figure 12. Visualizing features for APPROVE test set samples using tSNE. (a) Top level classes, math and literacy, are disjoint in feature space. (b-g) Since APPROVE is a multi-label dataset, we show one-vs-all tSNE plots.

C. Learned Feature Representation

We visualize learned features from our model using t-SNE in Figure 12. At the top level literacy and math videos are cleanly separated. We also plot one-vs-all plots for each class as APPROVE is a multi-label dataset. It can be observed that even fine-grained classes are well clustered, especially for math topics.
D. Implementation Details

D.1. Models

**Image Backbone:** We use ImageNet pretrained ResNet50 [20] and Instagram user generated tag weakly supervised (SWAG) + ImageNet finetuned ViT-B/16 (384×384 resolution) [18] from the TorchVision model zoo. We also use the ImageNet-21K pre-trained and ImageNet finetuned ViT-B/32 (224×224 resolution) from HuggingFace.

**Text Backbone:** For the language encoder, we use DistilBERT-Base-uncased [49], and t5-small [45] from the huggingface transformers library.

D.2. ASR Generation:

We generate ASR text from the videos using OpenAI-whisper [44], which is an open source ASR model. We used the medium.en version of the model and turned off the condition on previous text option as we observed that ASR generation would collapse for videos with poor audio quality with that option turned on by default.

D.3. Other Datasets

Some video classification datasets have been proposed with class labels based on topics being discussed or illustrated. One such dataset is COIN [56], which consists of instructional videos from 180 diverse coarse-grained tasks covering a wide range from changing-car-tire to making-pizza. While temporal sub-task segmentation labels are also available, in this paper we restrict ourselves to fine grained video classification task. YouTube-8M(YT-8M) [1] dataset is a large sample of YouTube data labeled with many coarse-grained visual entities, however, because of its large size, its distributed in the form of extracted visual and audio features. In order to fully test the potential of our method on this dataset, we create a subset using 1% of YT-8M data called YT-46K, which consists of 46,000 videos (note that despite its name the full YT-8M only contains 5.6 Million videos, since we scraped a 1% shard, we attempted scraping about 56,000 videos, of which about 46,000 were still available) with full video, audio and text metadata scraped from YouTube. Since it is a long tailed multi-label dataset, the frequency of labels follows a power-law distribution. We restrict the number of classes to those with at least 100 instances, which results in 166 usable labels.
D.4. Data Augmentations

We use RandAugment and RandomResizedCrop for augmenting video frames. RandAugment magnitude is ramped up from 1 to 10 over first 20 epochs.

```python
import torchvision.transforms as t

t.Compose([t.RandAugment(magnitude=magnitude),
    t.RandomResizedCrop(224, scale=(0.4, 1.0), 
    interpolation=t.InterpolationMode.BILINEAR),
    t.ConvertImageDtype(torch.float32),
    t.Normalize((0.5, 0.5, 0.5), (0.5, 0.5, 0.5)),
])
```

For augmenting text data we use synonym replacement from paraphrase dataset, random span cropping and random word swapping. As Back Translation is computationally expensive, we compute 4 additional back translated versions of each text before training.

```python
import nlpaug.augmenter.word as naw
import nlpaug.flow as naf

# m represents the magnitude of augmentation
m = 0.5

t = [naw.SynonymAug(aug_src="ppdb", aug_min=2, aug_max=15, aug_p=m)]
t += [naw.RandomWordAug(action="crop", aug_min=1, aug_max=5, aug_p=m)]
t += [naw.RandomWordAug(action="swap", aug_min=1, aug_max=5, aug_p=m/2.)]
train_augmenter = naf.Sequential(t)

from nlpaug.augmenter.word \ .back_translation import BackTranslationAug as BTAug

# actual arguments to BTAug are from_model_name, to_model_name
# abbreviated to fit the command in one line
BTAug(from='facebook/wmt19-en-de', to='facebook/wmt19-de-en')
BTAug(from='Helsinki-NLP/opus-mt-en-de', to='Helsinki-NLP/opus-mt-de-en')
BTAug(from='Helsinki-NLP/opus-mt-en-nl', to='Helsinki-NLP/opus-mt-nl-en')
BTAug(from='Helsinki-NLP/opus-mt-en-fr', to='Helsinki-NLP/opus-mt-fr-en')
```


