Supplementary Material FunnyNet: Audiovisual Learning of Funny Moments in Videos

Zhi-Song Liu*,[1], Robin Courant*,[2], and Vicky Kalogeiton[3][0000−0002−7368−6993]

[1] Caritas Institute of Higher Education, Hong Kong
zhisong.liu@connect.polyu.hk
[2] VISTA, LIX, Ecole Polytechnique, IP Paris
{robin.courant, vicky.kalogeiton}@lix.polytechnique.fr
http://www.lix.polytechnique.fr/vista/projects/2022_accv_liu

In this supplementary material, we first discuss the ethical and societal impacts of our work (Section 1). Next, we cover more details of the proposed FunnyNet implementation, evaluation and analysis (Section 2) and then we give more details for our proposed unsupervised laughter detection pipeline (Section 3). Finally, we analyse the several video materials that accompany this submission (Section 4).

1 Ethical and Societal Discussion

Practical Impact. There are various potential applications for FunnyNet. First, it may be useful to collect a large dataset of funny moments, so that, for example, cognitive researchers could study funniness mechanisms at large scale. Next, it may be useful to enable artists to edit films more easily, without relying on live audience. Finally, it may be useful to enhance human-machine interactions. For instance, adding a sense of humour to conversational agents would make the relation more natural and spontaneous.

However, FunnyNet is part of artificial intelligence systems that tend to analyse complex human specificities and behaviors (e.g., conversational agents). And, as all these kinds of systems, we should be careful to their uses. For instance, in the particular case of FunnyNet, it could enhance identity fraud methods, by better mimicking the sense of humor of victims.

Societal Impact. FunnyNet is trained mainly with Western cultural materials, especially from the USA, which do not necessarily represent uniform demographics. In particular, we mainly tackle funniness in American sitcoms, which covers a very specific type of humour. Therefore, without fine-tuning, FunnyNet might have difficulties to generalize to funny moments from other cultures, as humour is highly thematic, and themes vary from a culture to another. Moreover, the audio modality might also be highly impacted by cultural bias, as expressiveness is strongly related to culture, e.g., actor performances change a lot from

* These authors contributed equally to this work.

a country to another, leading to misinterpretations. In addition to the cultural barrier, FunnyNet includes language bias. Indeed, the audio as well as the textual modality - even though FunnyNet does not rely on subtitles -, are trained with the English language. This can be a limiting factor for generalization and transferability across languages, as jokes or puns often rely on language specificities.

We also note that the textual modality is also limited by alphabets that vary among languages.

Environmental Impact. All experiments are done on two NVIDIA RTX2080 GPUs, with each of them requiring 215W in power supply. For this project, we use approximately 800 GPU hours. Training a FunnyNet model with all three modalities requires around 6 GPUs hours, which amounts to 1.29 kWh and 300.75g of CO2 emitted.

2 FunnyNet

2.1 Implementation and Evaluation Details

Implementation Details. The input audio is first downsampled by fixed sampling frequency (16000 Hz) then transformed to log-scaled Mel spectrogram by mel-spaced frequency bins F = 64. We train FunnyNet using Adam optimizer with a learning rate of 10^{-4} and a batch size of 32 using Pytorch [6]. At training, we use data augmentation both for visual and audio data. For the frames, we randomly apply rotation and horizontal/vertical flipping, and we randomly set the frame sampling rate to select the 8 frames for training. For the audios, we apply random forward and backward time shifts and random Gaussian noises. For subtitles, we tokenize them as $max\_length = 64$ inputs and send to the BERT model.

Datasets. **Friends** [7,8] contains all 25 episodes (∼23 minutes) from the third season of Friends (∼10 hours). We split the dataset in 15 training (1-15), 5 validation (16-20) and 5 test episodes (21-25). Each episode comes with video, audio, face, body, voice tracks and features with speaker identifiers. As mentioned in Section 4 in the main manuscript, we enrich this dataset by providing manually annotated laughter time-codes. These annotated time-codes consist of time-stamps of the start and the end of all canned or not laughter. This process results in 3.5k time-codes, with an average duration 3 sec (0.3-16.5 sec), 138 average number of laughter per episode (109 to 182). We will make these annotations available online upon acceptance.

Manual and Automatic Labels. For fair evaluation, at training we use the automatically detected laughter (Section 3.3 in the main paper) extracted by our proposed laughter detector, whereas at test time we experiment with both manual and automatic laughter. The comparison is shown in Table 2, where we report as 'Manual Labelling' the results on the test manually-labelled test set,

Table 1: Comparison to the state of the art on five datasets for all metrics. For all methods we indicate the modalities used A: audio, V: visual frames, F: Face, T: text. † Reproduced results: we use the exact model as in [1] (see Section 2.2)

Method / Metrics	TBBT				MHD			
	Pre	**Rec**	**F1**	**Acc**	**Pre**	**Rec**	**F1**	**Acc**
Random	43.1	49.9	46.3	50.0	61.4	51.7	56.1	50.9
All positive	43.2	100.0	60.3	43.2	60.8	100.0	75.6	60.8
All negative	0.0	0.0	0.0	56.8	0.0	0.0	0.0	39.2
MUStARD 2019 (V+T+A) [2]	-	-	-	-	-	-	-	-
MSAM 2021 (V+T) [3]	-	-	-	-	-	-	81.3	72.4
MISA 2020 (V+A+T) [4]	-	-	-	-	-	-	-	-
HKT [5]	-	-	-	-	-	-	-	-
LaughM† 2021 (T) [1]	67.4	61.3	64.2	70.5	77.8	**97.3**	**86.5**	76.3
FunnyNet: V+F+A	**70.3**	**68.8**	**69.6**	**74.0**	**84.7**	83.2	84.0	**79.3**
FunnyNet: V+A+T	69.3	79.0	73.8	75.8	78.9	88.3	83.4	78.6
FunnyNet: V+F+T	**76.7**	75.4	**76.0**	69.5	69.8	83.2	75.9	69.8
FunnyNet: V+F+A+T	73.0	**79.1**	75.9	**78.3**	83.4	87.2	85.2	**79.6**

Method / Metrics	MUStARD				UR-FUNNY				Friends			
	Pre	**Rec**	**F1**	**Acc**	**Pre**	**Rec**	**F1**	**Acc**	**Pre**	**Rec**	**F1**	**Acc**
Random	48.7	47.8	48.3	48.7	50.1	50.2	50.2	50.2	51.0	51.0	51.0	51.0
All positive	50.0	100.0	66.7	50.0	50.7	100.0	75.4	50.7	50.0	100.0	66.7	50.0
All negative	0.0	0.0	0.0	50.0	0.0	0.0	0.0	49.3	0.0	0.0	0.0	50.0
MUStARD 2019 (V+T+A) [2]	-	-	71.7	71.8	-	-	-	-	-	-	-	-
MSAM 2021 (V+T) [3]	-	-	-	-	-	-	-	-	-	-	-	-
MISA 2020 (V+A+T) [4]	-	-	-	62.2	-	-	-	69.8	-	-	-	-
HKT [5]	-	-	-	79.4	-	-	-	77.4	-	-	-	-
LaughM† 2021 (T) [1]	69.0	68.7	68.6	68.7	62.8	**84.1**	71.9	67.6	59.9	**99.1**	74.7	59.8
FunnyNet: V+F+A	**79.7**	**83.2**	**81.4**	**81.0**	**90.1**	78.1	**83.7**	**78.0**	**85.8**	91.9	**86.8**	**84.8**
FunnyNet: V+A+T	**81.0**	78.0	79.5	79.9	89.1	79.6	84.1	79.9	85.2	91.4	88.2	85.8
FunnyNet: V+F+T	76.2	74.2	75.2	76.3	84.3	80.4	82.3	73.0	81.7	80.9	81.3	76.2
FunnyNet: V+F+A+T	78.3	**88.7**	**83.2**	**82.0**	86.1	82.8	**84.4**	**80.2**	84.6	93.4	**88.8**	**86.4**

and as 'Automatic Labelling' the results on the test set where the labels come from our Unsupervised laughter detector.

Table 2 shows that FunnyNet performs similarly on the 'Manual' and 'Automatic' labelling cases. This indicates that the laughter detected by our proposed automatic labelling model matches with human labelling level. Specifically, some results with automatic labels are higher than the manual ones, as they are not perfect, and probably some difficult cases are not detected; yet, overall, both results are comparable, showing that our laughter detector is sufficiently accurate.

2.2 Comparison to the State of the Art

Table 1 in the main manuscript reports the comparison to the state of the art on all five datasets. Table 1 here, reports the same comparison by including two additional metrics: Precision (Pre) and Recall (Rec). Note that most methods do not provide full metrics but only F1 score or accuracy, so there are some missing results. We report precision and recall to let the readers know that the

Table 2: Ablation study of FunnyNet on Friends. We experiment on both the manually (a) and the automatically (b) labelled test sets. We report Precision (Pre), Recall (Rec), F1 score (F1) and Accuracy (Acc) for audio (A), visual (V) and face (F) modalities

Modalities			Metrics			
A	V	F	Pre	Rec	F1	Acc
✓	-	-	71.16	75.09	73.71	66.73
-	✓	-	72.57	73.23	72.93	63.44
-	-	✓	72.58	73.22	72.94	62.13
✓	✓	-	80.86	87.81	84.19	81.14
✓	-	✓	81.13	86.89	83.91	82.22
✓	✓	✓	85.09	88.56	86.79	84.75

(a) Manual labelling

Modalities			Metrics			
A	V	F	Pre	Rec	F1	Acc
✓	-	-	70.23	75.87	72.94	67.80
-	✓	-	73.08	74.49	73.77	65.01
-	-	✓	74.21	74.09	74.15	64.23
✓	✓	-	81.11	88.20	84.51	82.19
✓	-	✓	81.67	87.39	84.43	82.28
✓	✓	✓	86.30	89.44	87.84	85.34

(b) Automatic labelling

improvement of using multiple modalities is consistent with the results of F1 score and accuracy.

As described in Section 5.1 in the main manuscript, performances of FunnyNet are noticeably better than of all other methods across the five datasets. We note that LaughM leads to higher recalls, e.g. by 14.1-6% on MHD and UR-FUNNY, which results in a higher F1 score only on MHD by 2.5%. However, these high recall values are accompanied by lower precisions in comparison to FunnyNet, e.g. by 27.3-25.9% on UR-FUNNY and Friends, showing that several predictions of LaughM are false positives. Moreover, we observe that adding the textual modality to FunnyNet improves the recall, while keeping the same precision, e.g. by 10.3-5.5% on MHD and MUStARD for FunnyNet V+F+A+T.

LaughM [1] (T) Results. LaughM [1] reports results on TBBT. For the MHD, MUStARD, UR-FUNNY and Friends datasets, we run the LaughM model and produce their results using the Punch model (only text), as it is the one that performs the best in their paper.

LaughM (T) [1] is trained and tested on TBBT. The protocol of LaughM consists in predicting a humour label for each consecutive subtitle of an episode. Specifically, LaughM models a sequence by processesing five subtitles at the same time (the one from the current timestamp and the four previous ones) using LSTM cells.

In our case, we first pre-train LaughM first on Friends using their protocol and then fine-tune on the other datasets. For testing, we apply the pre-trained or fine-tuned model over all subtitles to exploit the context information (processed by the LSTM). For a fair comparison to FunnyNet, we keep only the prediction of the last subtitle in the 8-sec window, so that we evaluate both the LaughM and FunnyNet models on the exact same test set.

Table 3: to the state of the art of FLOPs count (FLOPs), number of parameters (Params) and inference runtime average (Runtime)

Model	FLOPs (10^9)	Params (10^6)	Runtime (ms)
MISA 2020 (V+A+T) [4]	138.8	111.2	33.64
HKT 2021 (V+A+T) [5]	7.6	16.8	25.91
LaughM 2021 (T) [1]	2.5	112.4	11.15
FunnyNet (V+F+A)	190.9	126.9	45.25

For Friends, we use as positive subtitles the ones in the 8-sec window proceeding the laughter and negative all other (similar to Friends).

Then, we fine-tune the model on each dataset at hand and test it on the test set of each of them. For MHD, as we do not have training data, we fine-tune the model with 32 episodes from TBBT (disjoint set from MHD). In contrast to TBBT where subtitles from one episode are fed consecutively, here, we consider each sample as an independent one: the LSTM operates only on one sequence of subtitles (with five dialogue turns per sample). Similar to Friends, in order to have the exact same test set as with FunnyNet, we run the model over all subtitles, but we only keep the prediction on the last dialogue in the sample.

For MUStARD and UR-Funny, we fine-tune LaughM on the whole datasets. Note, we only report results for the punchline, as it leads to better performances. For these datasets, there is no context: each sample corresponds to one subtitle. For fair comparison, for MUStARD, we evaluate the model over the 5-folds provided by the authors of [2]. We also compare in Table 3 FunnyNet's complexity to the other state-of-the-art models. We note that the gain in performances and the unsupervised aspect aspect of FunnyNet impact its complexity. Indeed, FunnyNet is a huge model, with an increase of $\sim$ 52 GFLOPS, $\sim$ 16M of parameters and $\sim$ 11ms on runtime, in comparison to the second heaviest model [4].

2.3 Additional Ablations

Influence of Time Window Settings Time Window Length. Our proposed FunnyNet is trained on 8-second inputs of multiple modalities. This setting is based on the pre-trained Timesformer [9]. Since it processes video sequences of fixed length, we follow the same strategy to process the video frames. In order to test the effect that the length of time window has on FunnyNet, we perform here an ablation study on using Timesformer for funny prediction; this is equivalent to FunnyNet: V in our results of Table 3 in the main manuscript.

For this, we use input time windows of varying lengths (from 2 to 16 seconds) in FunnyNet: V and report the results on four datasets (as well as their average) in Table 4 and Figure 1 (a). Overall, we observe that using 8 seconds achieves the best performance compared to all other settings. Using 16 or 32 seconds input are the two follow-up scenarios, whereas having longer inputs degrades the performance. This is probably because longer inputs contain too many visual or

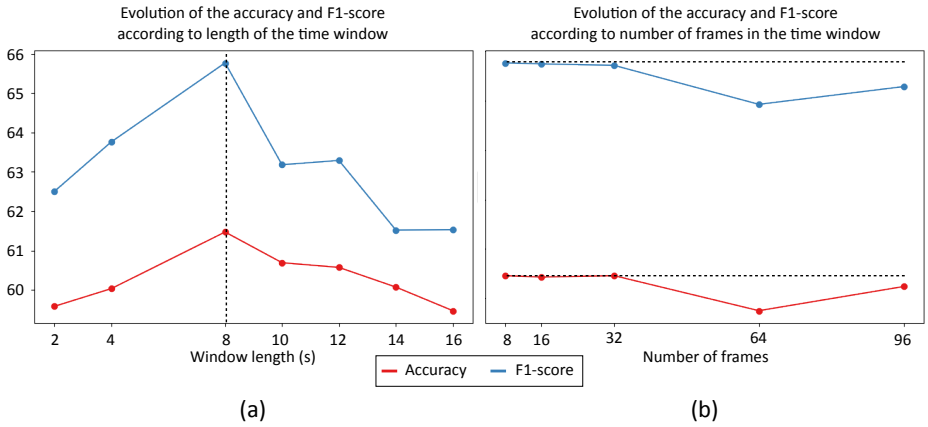

Fig. 1: FunnyNet performance over (a) varying lengths of time windows and (b) numbers of frames averaged across 4 datasets: TBBT, MHD, MUStARD and Friends. Prediction results when (a) varying the length of input time windows, with same 8 sampled frames, (b) using different numbers of frames within the same 8-second time window. We display Accuracy (*red*) and F1-score (*blue*)

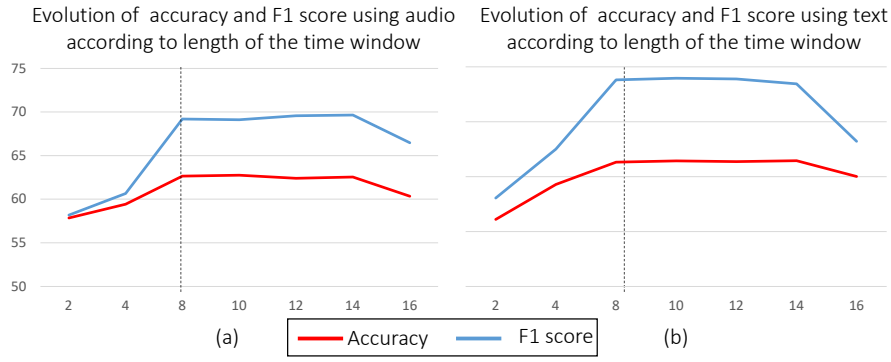

Fig. 2: FunnyNet performance over (a) varying lengths of time windows on audio modality and (b) varying lengths of time windows on text modality averaged across 4 datasets: TBBT, MHD, MUStARD and Friends. Prediction results We display Accuracy (*red*) and F1-score (*blue*)

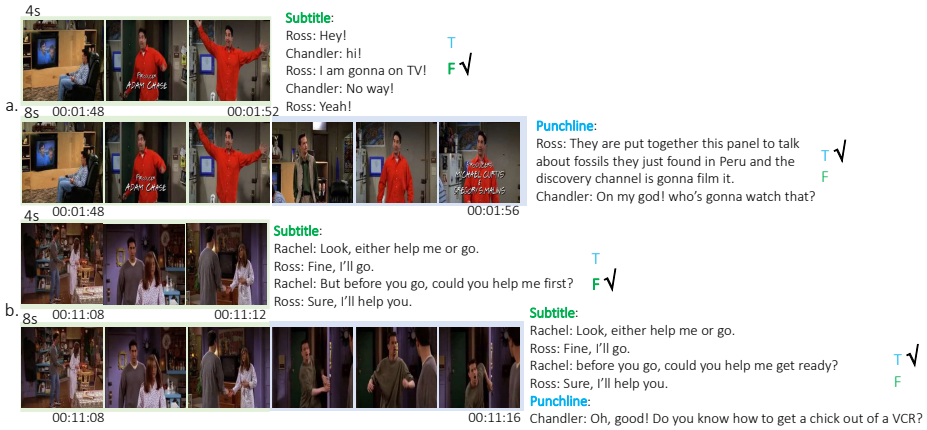

Fig. 3: Ablation study of FunnyNet on time windows smaller than 8 seconds. We show two examples cropped at different timestamps. The predictions are indicated by ✓

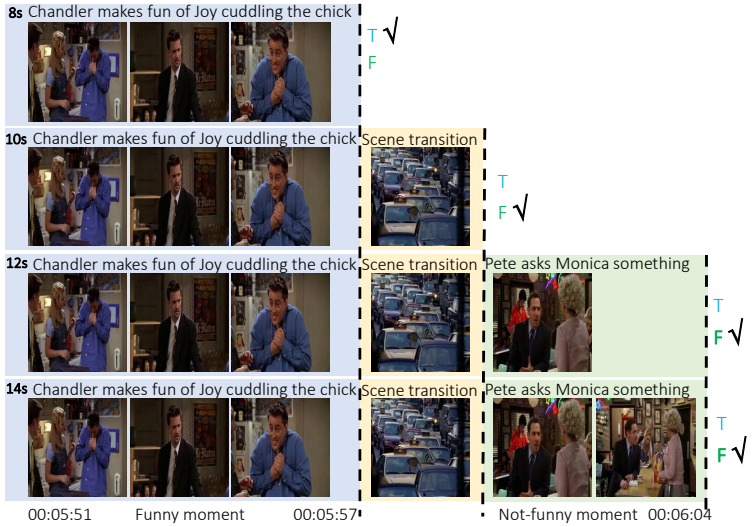

Fig. 4: Ablation study of FunnyNet on time windows larger than 8 seconds. We show 4 input examples cropped at different timestamps. Blue color indicates the funny moment, green color indicates not-funny moment and yellow color is for the scene transition. The predictions are indicated by ✓

Table 4: Ablation study of FunnyNet: V on different datasets on time windows. Prediction results on using different lengths of time windows with same 8 sampled frames

Time window	TBBT		MHD		MUStARD		Friends		Avg	
Metric	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc
2s	60.2	60.5	60.5	61.9	68.5	59.9	60.9	56.0	62.5	59.6
4s	60.2	60.7	60.4	62.3	68.4	59.3	66.0	57.8	63.8	60.0
8s	60.6	60.4	60.8	62.0	68.8	60.0	72.9	63.4	65.8	61.8
10s	59.5	58.8	58.8	60.7	68.8	60.7	65.7	62.7	63.2	60.7
12s	58.4	58.6	59.3	60.3	69.3	60.3	66.1	63.1	63.3	60.6
14s	58.5	57.7	58.4	59.8	68.4	59.8	60.8	63.0	61.5	60.1
16s	58.3	57.3	58.4	59.7	68.4	59.7	61.0	61.1	61.5	59.5

Table 5: Ablation study of FunnyNet: A on different datasets on the length of time window. Prediction results on using different time windows

Time window	TBBT		MHD		MUStARD		Friends		Avg	
Metric	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc
2s	57.1	57.3	56.8	57.8	59.8	58.1	59.1	58.2	58.2	57.9
4s	59.9	59.0	58.1	59.3	63.3	60.0	61.3	59.4	60.6	59.4
8s	67.2	62.3	65.9	60.6	70.0	61.0	73.7	66.7	69.2	62.6
10s	67.2	62.5	65.7	61.0	70.0	61.0	73.6	66.6	69.1	62.8
12s	67.2	62.1	65.6	60.2	70.1	60.8	75.3	66.5	69.6	62.4
14s	67.9	62.3	65.5	60.4	71.5	61.2	73.8	66.3	69.7	62.5
16s	67.8	62.2	65.4	60.4	70.3	59.9	62.4	58.9	66.5	60.3

audio information across both positive and negative samples, which confuses the model to give correct predictions.

In the meantime, we also test the impact of time windows on audio and text modalities. As for the vision modality, we evaluate the performances on four datasets and report the results in Table 5 and Table 6. For audio, we observe that using a longer time window improves the prediction accuracy. The best time window setting is around 8s 12s. For any time windows outside this range, the performance is getting worse. Similarly for text, the best performance is around 8s. Both audio and text share the same trend, showing that they are linearly correlated to the time windows. It is also reasonable since audio and text contains similar cues for scene understanding. The overall evaluation on audio and text are shown in Figure 2.

To further demonstrate the effect of time window, we visualize in Figure 4 an example from Friends of a video with varying length of time window, i.e., the same scene in 8-sec, 10-sec, 12-sec, and 14-sec (first to fourth row, respectively). The 8-sec input showxs that the proposed FunnyNet correctly identifies the scene

Table 6: Ablation study of FunnyNet: T on different datasets on the length of time window. Prediction results on using different time windows

Time window	**TBBT**		**MHD**		**MUStARD**		**Friends**		**Avg**	
Metric	**F1**	**Acc**	**F1**	**Acc**	**F1**	**Acc**	**F1**	**Acc**	**F1**	**Acc**
Metric	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc
2s	56.6	50.4	56.8	57.8	59.8	58.1	59.0	58.2	58.1	56.1
4s	60.7	58.3	60.0	59.8	66.0	59.9	63.3	59.1	62.5	59.3
8s	68.2	61.6	66.7	59.8	70.2	59.5	70.1	64.3	68.8	61.3
10s	67.8	61.7	66.8	60.2	71.0	59.6	70.2	64.3	69.0	61.4
12s	67.8	61.6	66.7	60.2	71.3	59.6	69.7	64.1	68.9	61.4
14s	67.8	61.7	66.5	60.1	69.8	60.1	69.7	64.0	68.4	61.5
16s	61.9	59.8	60.8	60.0	62.8	59.5	67.3	60.8	63.2	60.0

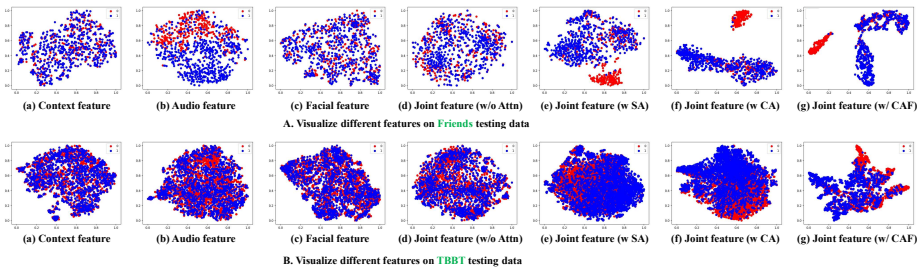

Fig. 5: t-SNE visualization of feature embeddings on Friends (*top*) and TBBT (*bottom*) for (*a*) visual, (*b*) audio, (*c*) face, (*d*) joint feature (V+A+F) without using any attention modules, (*e*) joint feature (V+A+F) using only self attention module (SA), (*f*) joint feature (V+A+F) using only cross attention module (CA), and (*g*) joint feature (V+A+F) using CAF module (both CA and SA, i.e., the proposed FunnyNet). We positive (*blue*) and negative (*negative*) samples

as a funny moment (given the smiling faces and their corresponding audio-track). However, increasing the length of time windows (from 8 to 14 secs) results in a scene transition and in turn a new scene that in this case is not funny (green blocks in Figure 4). Therefore, these models miss the funny moment, and instead wrongly predict the whole sequences as negative. This example shows that the length of the input window may have a strong impact on the prediction, as it may cross multiple scenes, and hence give mixed signals to the model (both funny and not-funny scenes), rendering it incapable of correct predictions.

Similarly, using a shorter time window may not be sufficiently long to capture the whole funny moment (e.g., miss the punchline). Figure 3 illustrates two such examples, where we observe that an input window shorter than 8-secs may not grasp the funny components, like the punchline, hence the model cannot give correct prediction.

Table 7: Ablation study of FunnyNet: V on different datasets on number of frames. Prediction results on using different numbers of frames with same 8-second time window

Data 8s	TBBT		MHD		MUStARD		Friends		Avg	
Metric	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc
8 frames	60.6	60.4	60.8	62.0	68.8	60.0	72.9	63.4	65.8	61.5
16 frames	60.5	60.4	60.8	62.1	68.8	59.9	73.0	63.3	65.8	61.4
32 frames	60.9	60.4	60.7	62.0	68.8	60.0	72.9	63.4	65.7	61.5
64 frames	60.3	60.2	59.8	60.0	68.7	60.0	70.9	62.8	64.9	60.8
96 frames	60.5	60.4	60.4	61.3	68.8	60.1	71.5	63.2	65.3	61.2

Thus, we conclude that the length of the input time window is a trade-off between too short (that do not always capture the funny-moment) and too long windows (that may span over multiple scenes). This trade-off typically depends on the dataset and on the type of funnyness we try to capture (e.g., punchlines vs black humor). In our experiments, we use 8-sec inputs, since on average, they perform the best among all alternatives (Table 4, Figure 1 (a)).

Number of Frames per Window. Given the input time window of 8 seconds, we also test another scenario when sample different number of frames in a fixed 8-second time window. The results are shown in Figure 1 (b) and in Table 7. We observe that the number of frames does not affect significantly the final prediction. This is expected, since sampling more frames in a fixed time window only produce redundancy without introducing new information.

Influence of Modalities

Feature Visualization. In Figure 6 in the main paper, we illustrate the t-SNE visualization of various feature embeddings. Figure 5 here displays more comparisons of different feature maps, with the top row displaying results on Friends and the bottom on TBBT. We display features both from single and from multiple modalities. We observe that when using a single modality, we cannot distinguish the positive from the negative samples ((a) to (c)). When using multiple modalities, FunnyNet learns to generate clearer boundaries for classification ((e) to (g)). For instance, the positive (dots) and negative (blue dots) are much separated than with using single modality. This pattern is clearer in the Friends dataset than in TBBT because TBBT has shorter and more frequent laughter. The model tends to identify the short clips ($\leq 2s$) as false positive. This demonstrates the powerful capabilities of using visual, audio and facial cues for funny moment prediction.

Cross Attention Fusion (CAF) Module. Figure 5 illustrates the t-SNE embeddings with positive and negative samples. (d) illustrates the embedding when

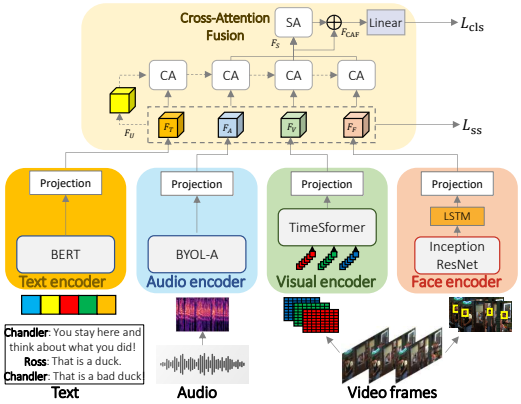

Fig. 6: FunnyNet architecture with Text Encoder. We use BERT to extract text features to combine with other modalities for prediction

Table 8: Ablation of loss of FunnyNet on Friends when training with and without the self-supervised contrastive loss L_{ss}

	Pre	**Rec**	**F1**	**Acc**
FunnyNet + L_{cls}	69.32	74.17	70.88	67.97
FunnyNet + L_{cls} + L_{ss}	**85.09**	**88.56**	**86.82**	**84.75**

training without the Cross Attention Fusion (CAF) Module, i.e., no self and cross attentions, while the last one has the t-SNE embedding when training the full FunnyNet with proposed CAF module. This qualitative comparison highlights the importance of the Attention mechanism for funny moment prediction, where we observe the significant change in the feature space and the clearly visible separability of positive from negative samples when using the Cross Attention Fusion module.

Text Encoder. Our proposed FunnyNet relies on audiovisual data for prediction without using text (in the form of subtitles) as extra information. As mentioned in Sections 3 and 6.1 in the main manuscript, for a fair comparison to the state of the art, we add a text encoder (that uses subtitles) as a fourth modality and train FunnyNet with a combination of all four modalities. Figure 6 illustrates this. Our text encoder consists of a pre-trained BERT model [10] followed by LSTM to extract text features for prediction. The input of the model is sequential sentences, which are then projected onto a fixed 768-D embedding vector using a pre-trained BERT. This vector is further processed by a LSTM, forming finally a 512-D vector $\mathbf{F}_{\text{T}} \in \mathbb{R}^{512}$.

Losses. FunnyNet uses the classification L_{cls} and the self-supervised contrastive losses L_{ss}. Here, we examine their impact by training FunnyNet with and without

Table 9: Modality importance. We test FunnyNet on triplets of data with one or more modalities coming from another timestamp (*unsynced*) and the rest coming from the same timestamp (*synced*)

Synced	Unsynced	Pre	Rec	F1	Acc
A + V + F	-	81.01	90.28	85.40	80.01
A + V	F	75.10	80.13	77.64	66.23
A + F	V	60.67	50.09	56.70	56.93
V + F	A	54.55	36.00	43.37	53.00
V	A + F	56.23	33.79	43.29	52.45
A	V + F	65.71	82.00	72.01	62.19

L_{ss}. Table 8 reports the results on Friends, where we observe that adding L_{ss} improves over +10 in all metrics. This reveals that using the auxiliary self-supervised task of syncing audiovisual data actually helps to identify the funny moments in videos.

Recall that the total loss is defined as: $L{=}\lambda_{\text{ss}}L_{\text{ss}} + \lambda_{\text{cls}}L_{\text{cls}}$, where λ_{ss}, λ_{cls} are the weighting parameters that control the relative importance of each loss.

In our training, we set the weighting parameters $\lambda_{\text{ss}} = 0.1$ and $\lambda_{\text{cls}} = 1$.

Modality Importance. To examine the modality importance, we design the following experiment. We test FunnyNet on unsynced misaligned data, i.e., we create triplets of data, where some modalities (one or more) come from different timestamps and different class labels, e.g., the face and visual frames come from the same timestamp i with label 'positive', whereas audio comes from timestamp j with label 'negative' (fourth row). For this, we collect 100 samples (50 pos and 50 neg) from Friends. Table 9 reports the results. Note again that the results are different from Table 1 in the main manuscript because this test is performed only on a subset of Friends. The first row corresponds to the baseline, where we use all modalities correctly. The cases where audio-video or audio-face are correct (second and third rows, respectively) outperform the case where video and face are correct (fourth row), thus indicating that audio is more robust than the visual data (either frames or faces). Similarly, in rows 5-6, we observe that when only audio is correctly synced leads to better performance than when only frames are correctly synced, probably because funny moments are not always accompanied by grimaces or facial expressions or general global context, but they can be associated to speech. Overall, our findings show that audio plays the most important role for funny moment prediction, which corroborates our intuition that audio captures basic cues of funniness, such as tone, pauses, etc.

Influence of Fine-Tuning Recall that FunnyNet is pre-trained on Friends and fine-tuned on different datasets for evaluation. To show the importance of fine-tuning, we present in Table 10 quantitative results with (w/ FT) and without fine-tuning on the dataset at hand (w/o FT). Specifically, we report FunnyNet

Table 10: Comparison with or without fine tuning on different datasets. We show precision (*Pre*), recall (*Rec*), F1 score (*F1*) and accuracy (*Acc*) on different datasets.

Model		Metric	TBBT	MHD	MUStARD	URFUNNY
A+V	**w/o FT**	**Pre**	62.33	65.34	64.59	64.29
		Rec	64.09	68.99	65.08	64.79
		F1	63.20	67.12	64.83	64.54
		Acc	65.78	65.10	64.44	61.23
A+F		**Pre**	60.45	65.34	62.33	60.12
		Rec	61.90	68.09	64.49	60.09
		F1	61.17	66.69	63.39	60.11
		Acc	65.67	63.10	63.29	60.09
A+V+F		**Pre**	61.71	77.00	80.44	65.51
		Rec	65.27	70.12	66.89	60.23
		F1	63.44	73.40	73.04	62.76
		Acc	64.12	70.02	67.43	61.22
A+V	**w/ FT**	**Pre**	72.09	79.66	78.97	86.08
		Rec	68.54	80.15	79.12	76.34
		F1	70.27	79.90	79.04	80.92
		Acc	72.77	73.14	77.50	76.79
A+F		**Pre**	68.45	82.90	81.39	86.07
		Rec	72.33	78.32	79.33	80.11
		F1	70.34	80.54	80.35	82.98
		Acc	72.89	73.89	78.90	79.44
A+V+F		**Pre**	70.31	84.72	79.72	90.13
		Rec	68.83	83.19	83.19	78.09
		F1	69.56	83.95	81.42	83.68
		Acc	74.00	79.30	81.01	78.00

results with three cases of inputs: audiovisual data (A+V), audio and facial data (A+F), audiovisual and facial data (A+V+F).

We observe that fine-tuning improves the overall performance, since different datasets have very different funny labels and different types of humour. For instance, the TBBT and MHD datasets have rather dense funny labels, i.e., each testing video clip is much shorter than 8-secs. MUStARD and UR-FUNNY have very uneven samples, ranging from 1 to 15 seconds. All these are different to the Friends training setup, where FunnyNet is pre-trained. Fine-tuning FunnyNet enables it to learn the biases between different datasets, leading to better results.

However, for all datasets, even without fine-tuning the accuracy (Acc) is between 60% and 70% (+10-20% compared to random). This shows that using pre-trained models on the same task helps the network generalize better. Notably, the lowest accuracy is on UR-Funny, where for all input combinations is accuracy is barely above 60%. For instance, we have on UR-Funny 61.2% for A+V, 60.1% for A+F, and 61.2% for A+V+F vs 65.1% for A+V, 63.1% for A+F and 70.0% for A+V+F for MHD. This is expected due to the domain shift between UR-Funny and the other four datasets (TED talks vs sitcoms).

Table 11: Comparison of laughter detection feature extraction. We compare results of our laughter detector using two audio feature extractors and their combination

	Temporal				**Det** IoU = 0.3			**Det** IoU = 0.7		
	Acc	Pre	Rec	F1	Pre	Rec	F1	Pre	Rec	F1
Ours Wav2CLIP[11]	77.56	64.49	63.66	63.70	91.25	61.23	73.07	49.74	33.45	39.89
Ours BYOL-A[12]	**85.97**	**76.94**	**79.38**	**77.81**	**94.57**	**82.25**	**87.83**	**54.07**	**47.11**	**50.27**
Ours [11] + [12]	85.91	77.13	78.80	77.62	94.52	81.91	87.60	54.18	47.02	50.26

3 Unsupervised Laughter Detection

Laughter Detection Metrics. To evaluate our laughter detector, we use: (a) sample-scale at the detection level: we compute the IoU between each pair of predicted and ground-truth segments and consider true positives the samples with IoU greater than a threshold; (b) frame-scale at the temporal level: for each frame, we check if it is correctly predicted as laughter or not. Then, we compute precision, recall and F1 score.

Note, at the detection level, we cannot compute Acc as the true negatives cannot be defined since there are no negative predictions.

Influence of the Feature Extraction. For feature extraction, we measure the efficiency of two pre-trained encoders and their combination, i.e. concatenation of the two feature vectors: BYOL-A [12] and Wav2CLIP [11][3].

We report the results in Table 11. In addition to the analysis in Section 5.2 in the main manuscript, we note that combining the two audio representations does not increase our detector performances: BYOL-A [12] embedding contains all the important information.

Influence of Clustering on the Detection Performance. Our unsupervised laughter detector (Figure 3 in the main manuscript) takes raw waveforms as input to detect laughters. It consists of (i) removing voices by subtracting channels, (ii) detecting peaks, and (iii) clustering audios to music and laughter. Here, we examine the impact the number K of clusters from K-means has on the final laughter detection performance.

In our experiments, we observe that there are much more laughter chunks than music chunks. Hence, we consider the smallest obtained cluster as being the music cluster that we remove and keep the laughter cluster. Figure 7 shows the performance of the detection pipeline at the detection (red lines) and temporal (blue lines) level as a function of different number of clusters (x axis). We observe:

– For 1 cluster: Using one cluster is equivalent to no clustering.

[3] Inspired by [13], BYOL-A [12] learns audio representations in a self-supervised manner by minimizing a similarity loss between outputs of two different augmentations of the same input. Wav2CLIP [11] learns a robust audio representation by distilling from CLIP [14].

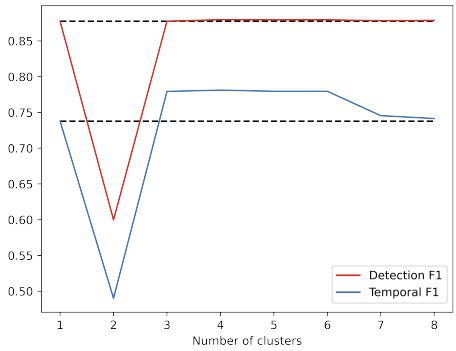

Fig. 7: Evolution of the temporal (*blue*) and detection (*red*) F1 scores according to the number of clusters chosen for the K-means algorithm at the end of the laughter detection pipeline

- For 2 clusters: we note that F1 scores reach a global minimum. Here, there is not enough degree of freedom for the K-means algorithm, which cannot detect the centroid of the music cluster as it is a small one.
- Between 3 and 6 clusters: we note that F1 scores are higher than for 1 cluster. Here, there are enough degrees of freedom for the K-means algorithm to correctly detect the centroid of the music cluster.
- Between 7 and $+\infty$ clusters: we note that F1 scores tend to return to the same value as for 1 cluster. Here, there are too many degrees of freedom for the K-means algorithm, and therefore it detects multiple centroids for the music cluster. Thus, the higher the number of clusters, the more small music sub-cluster we have, with the extreme case of having one cluster per sample, thus having the same effect as no clustering.

Moreover, Figure 7 shows that the detection F1 score (red line) is less sensitive to the number of clusters than the temporal F1 score (blue line). This can be explained by the fact that music chunks are generally longer than laughter chunks, and hence by removing longer false positive chunks we improve temporal metrics, whereas the impact is less important at the sample scale for detection metrics.

4 FunnyNet Videos

In the project page, we also include several videos that display the results when applying FunnyNet on videos from several domains: Friends and TBBT (sitcoms with canned laughter), Modern Family (sitcom without canned laughter) and also videos from other domains, i.e., movies, stand-up comedies, and audiobooks (see Section 6.3 in the main manuscript for more details).

Canned Laughter: FunnyNet on sitcoms with canned laughter and stand-up comedies with live audience laughter. We apply FunnyNet on both sitcoms with canned laughter and on a stand-up comedy with audience laughter. Videos 'FunnyNet - Sitcoms w/ laughter'[4] and 'FunnyNet - Stand-up comedy'[5] display the FunnyNet predictions on several clips from sitcoms (Friends and TBBT) and a Jerry Seinfeld stand-up comedy, respectively. We select funny moments thanks to the canned and audience laughter. We observe that FunnyNet correctly predicts the funny moments in videos. At the end of the videos, we also include some failure cases, which mostly contain dark frames or difficult context, like characters imitating laughter (see Section 6.2 in the main manuscript for more details).

No Laughter: FunnyNet on sitcoms and movies without canned laughter. We apply FunnyNet on the sitcom 'Modern Family', which does not contain canned laughter, and which we manually annotate (see Section 6.3 in the main manuscript for more details). We also apply FunnyNet on funny movie clips, that we select based on our own subjectivity, since they are not highlighted by laughter. Video 'FunnyNet - Sitcoms w/o laughter'[6] and 'FunnyNet - Movies'[7] show examples of Modern Family and movie scenes, respectively. We add canned laughter after each well detected funny moment. At the end of the videos, we also display some misclassified moments, for instance, we include one example of a scene with a character yelling at somebody else: situationally not funny, yet, contextually funny.

Funny Moments in Audio-Only Contents. The video 'FunnyNet - Sitcoms w/o laughter'[8] contains audio tracks of people saying jokes or reading a funny book, where we add canned laughter when a funny moment is well detected. We observe that FunnyNet predicts correctly the funny moments even without any visual cue. We also include a failure case where the joke punchline is not stressed enough.

[4] https://www.youtube.com/watch?v=6FHsm50Ie4g
[5] https://www.youtube.com/watch?v=0uwHNKAJ0kY
[6] https://www.youtube.com/watch?v=sb-bjW_gkj4
[7] https://www.youtube.com/watch?v=l5u2g1v8ua8
[8] https://www.youtube.com/watch?v=wuy2p_kPfpA

References

1. Kayatani, Y., Yang, Z., Otani, M., Garcia, N., Chu, C., Nakashima, Y., Takemura, H.: The laughing machine: Predicting humor in video. In: WACV. (2021) 3, 4, 5
2. Castro, S., Hazarika, D., Pérez-Rosas, V., Zimmermann, R., Mihalcea, R., Poria, S.: Towards multimodal sarcasm detection (an *Obviously* perfect paper). In: ACL. (2019) 3, 5
3. Patro, B.N., Lunayach, M., Srivastava, D., Sarvesh, S., Singh, H., Namboodiri, V.P.: Multimodal humor dataset: Predicting laughter tracks for sitcoms. In: WACV. (2021) 3
4. Hazarika, D., Zimmermann, R., Poria, S.: Misa: Modality-invariant and-specific representations for multimodal sentiment analysis. Proceedings of the 28th ACM International Conference on Multimedia (2020) 3, 5
5. Hasan, M.K., Lee, S., Rahman, W., Zadeh, A., Mihalcea, R., Morency, L.P., Hoque, E.: Humor knowledge enriched transformer for understanding multimodal humor. In: AAAI. (2021) 3, 5
6. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. In: NeurIPS. (2019) 2
7. Kalogeiton, V., Zisserman, A.: Constrained video face clustering using 1nn relations. In: BMVC. (2020) 2
8. Brown, A., Kalogeiton, V., Zisserman, A.: Face, body, voice: Video person-clustering with multiple modalities. In: ICCV. (2021) 2
9. Bertasius, G., Wang, H., Torresani, L.: Is space-time attention all you need for video understanding? In: Proc. ICML. (2021) 5
10. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: NAACL. (2019) 11
11. Wu, H.H., Seetharaman, P., Kumar, K., Bello, J.P.: Wav2clip: Learning robust audio representations from clip. arXiv preprint arXiv:2110.11499 (2021) 14
12. Niizumi, D., Takeuchi, D., Ohishi, Y., Harada, N., Kashino, K.: Byol for audio: Self-supervised learning for general-purpose audio representation. In: 2021 International Joint Conference on Neural Networks (IJCNN). (2021) 14
13. Grill, J.B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., Doersch, C., Avila Pires, B., Guo, Z., Gheshlaghi Azar, M., et al.: Bootstrap your own latent-a new approach to self-supervised learning. NeurIPS (2020) 14
14. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. Proc. ICML (2021) 14