REUSING PREPROCESSING DATA AS AUXILIARY SUPERVISION IN CONVERSATIONAL ANALYSIS

Anonymous authors
Paper under double-blind review

ABSTRACT

Conversational analysis systems are trained using noisy human labels and often require heavy preprocessing during multi-modal feature extraction. Using noisy labels in single-task learning increases the risk of over-fitting. However, including auxiliary tasks could improve the performance of the primary task learning. This approach is known as Primary Multi-Task Learning (MTL). A challenge of MTL is the selection of relevant auxiliary tasks that avoid negative transfer. In this paper, we explore how the preprocessed data used for feature engineering can be re-used as auxiliary tasks in Primary MTL, thereby promoting the efficient use of data in the form of auxiliary supervision learning. Our main contributions are: (1) the identification of sixteen relevant auxiliary tasks, (2) the method of distributing learning capacity between the primary and auxiliary tasks, and (3) the relative supervision hierarchy between the primary and auxiliary tasks. Extensive experiments on IEMOCAP and SEMAINE data validate the improvements over single-task approaches, and suggest that it generalizes across multiple primary tasks.

1 INTRODUCTION

The sharp increase in uses of video-conferencing creates both a need and an opportunity to better understand these conversations (Kim et al., 2019a). In non-real-time applications, analyzing conversation can give post-event feedback to improve communication skills (Hoque et al., 2013; Naim et al., 2015). In real-time applications, such systems can be useful in legal trials, public speaking, e-health services, and more (Poria et al., 2019; Tanveer et al., 2015).

Analyzing conversations requires both human expertise and a lot of time, which is the problem that many multimodal conversational analysis systems are trying to solve with automation. However, to build such systems, analysts often require a training set annotated by humans (Poria et al., 2019). The annotation process is costly, thereby limiting the amount of labeled data. Moreover, third-party annotations on emotions are often noisy. Noisy data coupled with limited labeled data increases the chance of overfitting (James et al., 2013).

From the perspective of feature engineering to analyze video-conferences, analysts often employ pre-built libraries (Baltrušaitis et al., 2016; Vokaturi, 2019) to extract multimodal features as inputs to training. This preprocessing phase is often heavy, and the features are used only as inputs. In this paper, we investigate how the preprocessed data can be re-used as auxiliary tasks in Primary MTL, thereby promoting the efficient use of data, in the form of auxiliary supervised learning. Specifically, our main contributions are (1) the identification of relevant auxiliary tasks, (2) the method of distributing learning capacity between the primary and auxiliary tasks, and (3) the relative supervision hierarchy between the primary and auxiliary tasks. We demonstrate the value of our approach through predicting emotions on two datasets, IEMOCAP (Busso et al., 2008) and SEMAINE (McKeown et al., 2011).

2 RELATED WORKS AND HYPOTHESES

Multitask learning has a long history in machine learning (Caruana, 1997). In this paper, we focus on Primary MTL, a less commonly discussed subfield within MTL (Mordan et al., 2018). Primary

MTL is concerned with the performance on one (primary) task – the sole motivation of adding auxiliary tasks is to improve the primary task performance.

In recent years, primary MTL has been gaining attention in computer vision (Yoo et al., 2018; Fariha, 2016; Yang et al., 2018; Mordan et al., 2018; Sadoughi & Busso, 2018), speech recognition (Krishna et al., 2018; Chen & Mak, 2015; Tao & Busso, 2020; Bell et al., 2016; Chen et al., 2014), and natural language processing (Arora et al., 2019; Yousif et al., 2018; Zalmout & Habash, 2019; Yang et al., 2019; Du et al., 2017). The addition of multiple tasks increases the risk of negative transfer (Torrey & Shavlik, 2010; Lee et al., 2016; 2018; Liu et al., 2019; Simm et al., 2014), which leads to many design considerations. Two of such considerations are, identifying (a) what tasks are relevant and (b) how much of the model parameters to share between the primary and auxiliary tasks. In addition, because we are performing Primary MTL, we have the third consideration of (c) whether we should prioritize primary supervision by giving it a higher hierarchy than the auxiliary supervision.

In contrast with previous works, our approach (a) identifies sixteen relevant auxiliary targets, (b) dedicates a primary-specific branch within the network, and (c) investigates the efficacy and generalization of prioritizing primary supervision across eight primary tasks.

Hypothesis H1: The introduced set of auxiliary supervision improves primary MTL.
We introduce and motivate the full set of sixteen auxiliary supervisions, all based on existing literature: these are grouped into four families, each with four auxiliary targets. The four families are (1) facial action units, (2) prosody, (3) historical labels, and (4) future labels:
(1) Facial action units, from the facial action coding system identifies universal facial expressions of emotions (Ekman, 1997). Particularly, AU 05, 17, 20, 25 have been shown to be useful in detecting depression (Yang et al., 2016a; Kim et al., 2019b) and rapport-building (Anonymous, 2021).
(2) Prosody, the tone of voice – happiness, sadness, anger, and fear – can project warmth and attitudes (Hall et al., 2009), and has been used as inputs in emotions detection (Garcia-Garcia et al., 2017).
(3 and 4) Features at different historical time-points (X_t, X_{t-1}, X_{t-2} etc.) is a common practice in statistical learning, especially in time-series modelling (Christ et al., 2018). Lastly, future laboratory test results are proven to be useful to train a pneumonia detection model (Caruana et al., 1996). Inspired by their work, we propose using historical (Y_{t-4}, Y_{t-3}, Y_{t-2}, Y_{t-1}) and future (Y_{t+4}, Y_{t+3}, Y_{t+2}, Y_{t+1}) target labels as auxiliary targets.

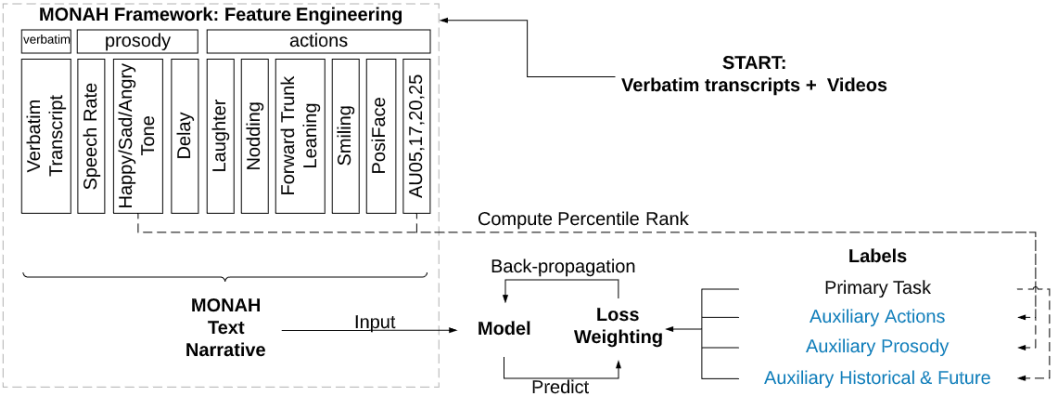

Figure 1: Reusing components (dotted lines) of the feature engineering process as auxiliary targets (in blue). The MONAH framework (Anonymous, 2021) is introduced in section 4.2

Given that the actions and prosody families have been proven useful as inputs, they are extracted as features. We propose to explore whether they can be reused as supervision (see Fig. 1). Our hypothesis **H1** is that re-using them as auxiliary supervision improves primary MTL. This is related to using hints in the existing MTL literature, where the auxiliary tasks promotes the learning of the feature (Cheng et al., 2015; Yu & Jiang, 2016).

Hypothesis H2: When the primary branch is given maximum learning capacity, it would not be outperformed by models with primary branch having less than the maximum learning

capacity.
Deeper models with higher learning capacity produce better results (Huang et al., 2019; Nakkiran et al., 2019; Menegola et al., 2017; Blumberg et al., 2018; Romero et al., 2014). Also, since the auxiliary branch is shared with the primary supervision, the auxiliary capacity should be limited to enforce information transfer and improve performance (Wu et al., 2020). Therefore, given a fixed learning capacity budget, our hypothesis **H2** is that we should allocate the maximum learning capacity to the primary branch because we care only about the primary task performance.

Hypothesis H3: Auxiliary supervision at the lower hierarchy yields better primary MTL as compared to flat-MTL.
Having the auxiliary tasks at the same supervisory level as the primary task is inherently sub-optimal because we care only about the performance of the primary task (Mordan et al., 2018). To prioritize the primary task, we could change the model architecture such that the auxiliary supervision is at the lower hierarchy than the primary supervision, which would be discussed in the next section.

3 Model Architecture

3.1 Flat-MTL Hierarchical Attention Model

We start with an introduction of the Hierarchical Attention Model (HAN) (Yang et al., 2016b). We chose HAN because of its easy interpretability as it only uses single-head attention layers. There are four parts to the HAN model, (1) text input, (2) word encoder, (3) talkturn encoder, and (4) the predictor. In our application, we perform our predictions at the talkturn-level for both IEMOCAP and SEMAINE. For notation, let s_i represent the *i*-th talkturn and w_{it} represent the *t*-th word in the *i*-th talkturn. Each single talkturn can contain up to *T* words, and each input talkturn can contain up to *L* past talkturns to give content context (to discuss in section 4.2).

Text Input Given a talkturn of words, we first convert the words into vectors through an embedding matrix W_e.

Word encoder The word encoder comprises of bidirectional GRUs (Bahdanau et al., 2014) and a single head attention to aggregate word embeddings into talkturn embeddings. Given the vectors x_{it}, the bidirectional GRU reads the words from left to right as well as from right to left and concatenates the two hidden states together to form h_{it}. We then aggregate the hidden states into one talkturn embedding through the attention mechanism. u_{it} is the hidden state from feeding h_{it} into a one layer perceptron. The attention weight given to u_{it} is the softmax normalized weight of the similarity between itself (u_{it}) and u_w, which are all randomly initialized and learnt jointly.

$$
\begin{aligned}
x_{it} &= W_e w_{it}, t \in [1, T] & u_{it} &= relu(W_w h_{it} + b_w) \\
\overrightarrow{h}_{it} &= \overrightarrow{GRU}(x_{it}), t \in [1, T] & \alpha_{it} &= \frac{exp(u_{it}^\top u_w)}{\Sigma_t exp(u_{it}^\top u_w)} \\
\overleftarrow{h}_{it} &= \overleftarrow{GRU}(x_{it}), t \in [T, 1] & s_i &= \Sigma_t \alpha_{it} h_{it} \\
h_{it} &= (\overrightarrow{h}_{it}, \overleftarrow{h}_{it})
\end{aligned}
$$

Talkturn encoder With the current and past talkturn embeddings (content context, to discuss in section 4.2), the talkturn encoder aggregates them into a single talkturn in a similar fashion, as shown below.

$$
\begin{aligned}
\overrightarrow{h}_i &= \overrightarrow{GRU}(s_i), i \in [1, L] & u_i &= relu(W_s h_i + b_s) \\
\overleftarrow{h}_i &= \overleftarrow{GRU}(s_i), i \in [L, 1] & \alpha_i &= \frac{exp(u_i^\top u_s)}{\Sigma_i exp(u_i^\top u_s)} \\
h_i &= (\overrightarrow{h}_i, \overleftarrow{h}_i) & v &= \Sigma_i \alpha_i h_i
\end{aligned}
$$

The simplest way of adding the sixteen (four auxiliary targets from four families of auxiliary supervision) auxiliary task predictors would be to append them to where the primary task predictor is, as

illustrated in Fig. 2. That way, all predictors use the same representation v. We refer to this architecture as flat-MTL because the auxiliary supervision is at the same level as the primary supervision. We are unable to test **H2** and **H3** using this architecture.

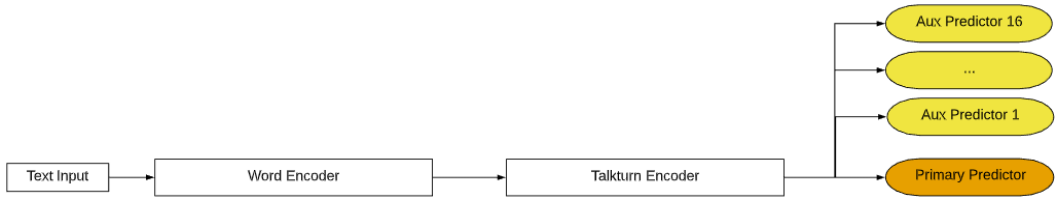

Figure 2: Forward pass of the Flat-MTL HAN architecture. Auxiliary tasks (yellow) are added at the same level of the primary task (orange).

3.2 HAN-ROCK

We adapted[1] the ROCK architecture (Mordan et al., 2018) which was built for Convolutional Neural Networks (LeCun et al., 1995) found in ResNet-SSD (He et al., 2016; Liu et al., 2016) to suit GRUs (Bahdanau et al., 2014) found in HAN (Yang et al., 2016b) (see Fig. 3).

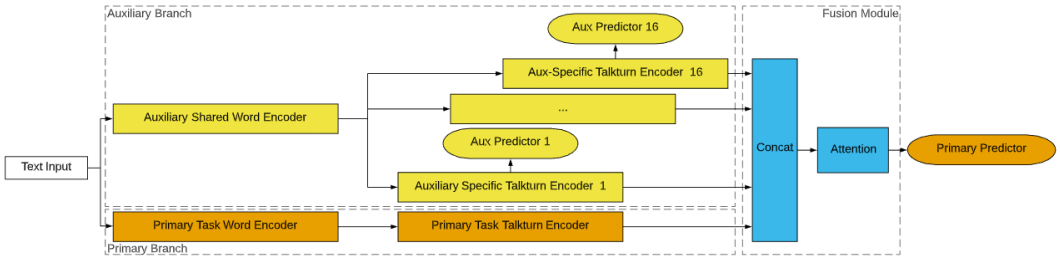

Figure 3: Forward pass of the HAN-ROCK architecture. There is a primary branch (orange) where auxiliary supervision (yellow) can not influence. The fusion module (blue) aggregates the talkturn embeddings from all tasks into one.

To study **H3**, we bring the auxiliary task predictors forward, so that the auxiliary supervision is of a lower hierarchy than primary supervision (see Fig. 3). It is of a lower hierarchy because the back-propagation from the primary supervision is able to temper the back-propagation from the auxiliary supervision but not vice-versa. This also sets us up to study **H2**, the impact of distributing different learning capacities to the auxiliary and primary branch. Each of the auxiliary tasks has its own talkturn encoder but shares one word encoder in the auxiliary branch (to keep the network small). Subscript a indicates whether the word encoder is for the primary or auxiliary branch:

$$x_{it} = W_e w_{it}, t \in [1, T] \qquad u_{it} = relu(W_{aw} h_{ait} + b_{aw})$$

$$\overrightarrow{h}_{ait} = \overrightarrow{GRU_a}(x_{it}), t \in [1, T], a \in \{pri, aux\} \qquad \alpha_{ait} = \frac{exp(u_{ait}^\top u_{aw})}{\Sigma_t exp(u_{ait}^\top u_{aw})}$$

$$\overleftarrow{h}_{ait} = \overleftarrow{GRU_a}(x_{it}), t \in [T, 1], a \in \{pri, aux\} \qquad s_a i = \Sigma_t \alpha_{ait} h_{ait}$$

$$h_{ait} = (\overrightarrow{h}_{ait}, \overleftarrow{h}_{ait})$$

Each task has its own talkturn encoder. Subscript b indicate which of the seventeen tasks – the primary talkturn task or one of the sixteen auxiliary tasks – is the talkturn encoder is dedicated to:

[1]Implementation available at https://github.com/anonymous/placeholder; Please see attached supplementary material for implementation details during the review phase.

$$\overrightarrow{h}_{abi} = \overrightarrow{GRU_{ab}}(s_{ai}), i \in [1, L], a \in \{pri, aux\}, b \in \{pri, aux1, aux2, ..., aux16\}$$
$$\overleftarrow{h}_{abi} = \overleftarrow{GRU_{ab}}(s_{ai}), i \in [L, 1], a \in \{pri, aux\}, b \in \{pri, aux1, aux2, ..., aux16\}$$

$$h_{abi} = (\overrightarrow{h}_{abi}, \overleftarrow{h}_{abi}) \qquad \alpha_{abi} = \frac{exp(u_{abi}^{\top} u_b)}{\Sigma_i exp(u_{abi}^{\top} u_b)}$$
$$u_{abi} = relu(W_b h_{abi} + b_b) \qquad v_{ab} = \Sigma_i \alpha_{abi} h_{abi}$$

The seventeen talkturn embeddings (v_{ab}) goes through a concatenation, then the single head attention, aggregating talkturn embeddings across seventeen tasks into one talkturn embedding for the primary task predictor. Subscript c pertains to the fusion module.

$$\text{concatenation: } v_c = (v_{ab}), a \in \{pri, aux\}, b \in \{pri, aux1, aux2, ..., aux16\}$$
$$\text{attention: } \alpha_c = \frac{exp(v_c^{\top} u_c)}{\Sigma_c exp(v_c^{\top} u_c)}$$
$$\text{overall primary talkturn vector: } v = \Sigma_c \alpha_c v_c$$

4 EXPERIMENTS

4.1 DATA AND PRIMARY TASKS

We validated our approach using two datasets with a total of eight primary tasks: the IEMOCAP (Busso et al., 2008) and the SEMAINE (McKeown et al., 2011) datasets. Both datasets are used in multimodal emotions detection research (Poria et al., 2019), and have fine-grained emotions annotations at the talkturn-level. We divided the datasets into train, development, and test sets in an approximate 60/20/20 ratio such that the sets do not share any speaker. Please refer to Appendix A.1 for details on the splits.

The target labels of the eight primary tasks are all at the talkturn-level. The four primary tasks of IEMOCAP consists of the four-class emotions classification (angry, happy, neutral, sad), and three regression problems – valance (1-negative, 5-positive), activation (1-calm, 5-excited), and dominance (1-weak, 5-strong). The primary tasks of SEMAINE consist of four regression problems – activation, intensity, power, valence. We use two standard metrics, mean absolute error (MAE), and 4-class weighted mean classification accuracy, MA(4), to evaluate the hypotheses.

4.2 INPUT

Multimodal feature extraction is computed using the MONAH framework (Anonymous, 2021). This framework uses a variety of pre-trained models to extract nine multimodal features, associated with the prosody of the speech and the actions of the speaker, and weaves them into a multimodal text narrative. We refer the reader to Anonymous (2021) for the details and efficacy of the MONAH framework. The benefit of the created narrative is that it describes what is said together with how it is said for each talkturn, giving richer nonverbal context to the talkturn (see Fig. 4 for an example).

Verbatim:	"The woman said no."
MONAH:	"The woman raised chin. The woman sadly and slowly said no."

Figure 4: Example of a MONAH transcript.

In addition to nonverbal context, we concatenated a variable number of preceding talkturns to the current talkturn as content context. Content context has been proven to be useful in DialogueRNN (Majumder et al., 2019). The content-context size is tuned as a hyperparameter. The resulting multimodal text narrative, consisting of both nonverbal and content context, is used as the sole input to the models. The impact of progressively removing nonverbal and content context is not a contribution of this paper and hence not analyzed.

4.3 AUXILIARY TARGETS

We first clarify the method of extraction for the auxiliary families. The OpenFace algorithm (Baltrušaitis et al., 2016) is used to extract the four continuous facial action units (AU) – AU 05, 17, 20, 25. The Vokaturi algorithm (Vokaturi, 2019) is used to extract the four continuous dimensions in the tone of voice – happiness, sadness, anger, and fear. As for historical and future features, we simply look up the target label for Y_{t-4}, Y_{t-3}, Y_{t-2}, Y_{t-1}, and Y_{t+4}, Y_{t+3}, Y_{t+2}, Y_{t+1}. If any of the historical or future labels is not available, for example, Y_{t-4} is not available for the third talkturn, we substitute with the next nearest non-missing label.

For all auxiliary targets that reused the input features (actions and prosody), we converted them into a percentile rank that has the range [0,1] using the values from the train partition. This is a subtle but note-worthy transformation. When reusing an input as an auxiliary target, it would be trivial if the input can easily predict the target. For example, given the following MONAH transcript as input, "The woman sadly and slowly said no." It would be trivial to use a binary (quantized) auxiliary target of "was the tone sad?" because we would only be training the model to look for the word "sadly". However, if the auxiliary target is a percentile rank (less quantized) of the sadness in tone, then the presence of the word "sadly" increases the predicted rank, but the model could still use the rest of the nonverbal cues ("slowly") and what is being said ("no") to predict the degree of sadness. That way, representations learnt for the auxiliary tasks uses more of the input. Additionally, this transformation also helped us side-step the decision to use the multimodal streams *exclusively* in the inputs or in the supervision (Caruana & De Sa, 1997; Caruana & de Sa, 1998) because we could use them in both.

Percentile rank also has the convenient property of having the range [0,1]. We scaled the percentile ranks so that they all have the same range as the primary task (see appendix A.2 for transformation details). This is done to ensure that if we assigned equal loss weights to all tasks, the contribution of every task (both auxiliary and primary) is of the same order of magnitude, an important consideration for MTL (Gong et al., 2019; Hassani & Haley, 2019; Sener & Koltun, 2018).

4.4 MODELS, TRAINING, AND HYPERPARAMETERS TUNING

The overall loss is calculated as the weighted average across all seventeen tasks: (1) we picked a random weight for the primary task from the range [0.50, 0.99], this ensures that the primary task has the majority weight. (2) For the remaining weights (1 - primary weight), we allocated them to the sixteen auxiliary tasks by: (a) random, (b) linearly-normalized mutual information, or (c) softmax-normalized mutual information. (a) is self-explanatory. As for (b) and (c), mutual information has been shown that it is a best predictor – compared to entropy and conditional entropy – of whether the auxiliary task would be helpful to primary MTL (Bjerva, 2017). We computed the mutual information (vector m) of each auxiliary variable with the primary target variable (Kraskov et al., 2004; Ross, 2014) using scikit-learn (Pedregosa et al., 2011). Then, we linearly-normalized or softmax-normalized m to sum up to 1. Finally, we multiplied the normalized m with the remaining weights from (2); this ensures that the primary weight and the sixteen auxiliary weight sum up to one. (a), (b), and (c) have ten trials each during hyper-parameters tuning.

Two variants of the HAN architectures are used (Fig 2 and 3). Glove word embeddings (300-dimensions) are used to represent the words (Pennington et al., 2014). Hyper-parameters tuning is crucial because different combinations of primary and auxiliary tasks require different sets of hyper-parameters. For hyperparameters tuning, we used random search (Bergstra & Bengio, 2012) with thirty trials (10 random sets of hyperparameters times three methods of allocating auxiliary loss weights). We tuned the learning rate, batch size, L2 regularization, the number of GRUs assigned to the primary and auxiliary branches, the auxiliary weights assignment, the content-context size, and lastly the GRU dropout and recurrent dropout. Appendix A.3 details the range of hyperparameters tuned.

Training is done on a RTX2070 or a V100 GPU, for up to 350 epochs. Early stopping is possible via the median-stopping rule (Golovin et al., 2017) after the fifth epoch and after every two epochs (i.e. at epoch number 5, 7, 9, ..., 349). Appendix A.4 details the hyperparameters of models that performed the best on the development set.

For hypotheses testing, we bootstrapped confidence intervals of the test set performance of both baseline and challenger models as well as the confidence intervals of the differences in performances. Please see appendix A.5 for details on the bootstrap procedure.

5 Results and Discussion

The key takeaways of the results are: (**H1**) The introduced set of auxiliary supervision improves primary MTL significantly in six of the eight primary tasks. (**H2**) Maximum learning capacity should be given to the primary branch as a default. (**H3**) HAN-ROCK is unlikely (in one of the eight tasks) to degrade primary MTL significantly, and sometimes significantly improves it (in four of the eight tasks).

(**H1**): To test **H1** (whether the introduced set of auxiliary supervision improves Primary MTL), we first train the model with all sixteen auxiliary targets (from families actions, prosody, historical, and future). Then we set the loss weights from historical and future targets to be zero; effectively, there is only supervision from eight auxiliary targets (actions and prosody). Lastly, for the baseline model (no auxiliary supervision), we set the loss weights from all sixteen auxiliary targets to zero.

Given auxiliary supervision, the model significantly outperforms the baseline of not having auxiliary supervision in six out of the eight primary tasks (Table 1). Comparing the baseline model with the model with two auxiliary target families, they significantly outperformed the baseline model in five out of eight primary tasks. The addition of two auxiliary target families (historical and future labels) sometimes significantly improved primary MTL (valence in IEMOCAP), but it also sometimes significantly made it worse (activation and intensity in SEMAINE). This shows that the value of auxiliary tasks, and the associated risk of negative transfer, depends on the auxiliary task.

Table 1: **H1** Results. *: the model performance has a statistically significant difference with the baseline model. a: action, p: prosody, h: historical labels, f: future labels.

	IEMOCAP				**SEMAINE**			
Aux. Target	**Classif. MA(4)**	**Val. MAE**	**Act. MAE**	**Dom. MAE**	**Act. MAE**	**Int. MAE**	**Pow. MAE**	**Val. MAE**
None (Baseline)	0.625	0.527	0.518	0.667	0.194	0.238	0.170	0.177
ap	0.715*	0.538	0.507	0.600*	0.184*	0.218*	0.167*	0.178
aphf	0.706*	0.497*	0.504	0.587*	0.187	0.231	0.165*	0.169

We found a supplementary finding whilst studying **H1**: the improvement on lower-resource (lower counts) classification labels is bigger than higher-resource labels (Table 2). There is a statistically significant improvement for the "sad" label, and no significant changes for the rest of the labels.

Table 2: Class-wise classification F1 score on IEMOCAP. Baseline (challenger) refers to HAN-Rock architecture without (with) auxiliary supervision. *: the challenger performance has a statistically significant difference with the baseline model.

Label	**Count**	**Baseline**	**Challenger**	**Improvement**
Sad	1,084	0.573	0.689*	0.116
Anger	1,103	0.531	0.683	0.152
Happy	1,636	0.772	0.784	0.012
Neutral	1,708	0.664	0.636	-0.028

(**H2**): To test **H2** (whether maximum learning capacity should be given to the primary branch), we let P represent the number of GRU assigned to the primary talkturn encoder, and A represent the number of GRU assigned to each of the sixteen auxiliary talkturn encoder. We constrained $P + A$ to be equal to 257. During our experiments, we set P to 1, 64, 128, 192, and 256. We set 256 as the baseline model because it is the maximum learning capacity we can give to the primary branch while giving 1 GRU ($= 257 - 256$) to each of the sixteen auxiliary talkturn encoders.

In all primary tasks, the baseline model of assigning 256 GRUs to the primary branch is not significantly outperformed by models that assigned 1, 64, 128, 192 GRUs (Table 3). Generally, the

performance decreased as the number of GRUs assigned to the primary talkturn encoder decreased from 256 to 1, and we observed significantly worse performance in two out of eight tasks – in power and valence in SEMAINE. Also, assigning 256 GRU to the primary talkturn encoders and 1 to each of the sixteen auxiliary talkturn encoders yields the smallest model[2], and thus trains the fastest. Therefore, we recommend that the maximum capacity be given to the primary branch as a default. That said, the presence of an auxiliary branch is still important. The baseline of **H1** (no auxiliary supervision, Table 1) can be approximated as P=$256 + 16 \times 1$, A=0 because the model architecture is the same, except that the loss weights of all auxiliary tasks are zero. We compared the former to the baseline in Table 3, and found that four out of eight primary tasks have significant improvements by changing the number of talkturn encoders assigned to each auxiliary task from zero to *one*.

Table 3: **H2** Results. *: the model performance has a statistically significant difference with the baseline model (P = 256). ˆ: Assigning 1 GRU to the auxiliary task talkturn encoder yields a statistically significant difference with assigning 0 GRU.

	IEMOCAP				**SEMAINE**			
Pri. GRU	**Classif. MA(4)**	**Val. MAE**	**Act. MAE**	**Dom. MAE**	**Act. MAE**	**Int. MAE**	**Pow. MAE**	**Val. MAE**
256 (Baseline)	0.715ˆ	0.538	0.507	0.587ˆ	0.187	0.231	0.165ˆ	0.169ˆ
192	0.736	0.509	0.518	0.604	0.187	0.228	0.165	0.184*
128	0.711	0.537	0.512	0.597	0.189	0.216	0.176*	0.196*
64	0.687	0.540	0.507	0.593	0.191	0.234	0.167	0.192*
1	0.656	0.554	0.509	0.599	0.190	0.229	0.168*	0.191*

(**H3**): To test **H3** (whether auxiliary supervision should be given a lower hierarchy), we compare the results from the flat-MTL HAN architecture (baseline) against the HAN-ROCK architecture (Table 4). Placing auxiliary supervision at the lower hierarchy significantly improves primary MTL in four out of eight tasks. In only one out of eight tasks (power in SEMAINE), auxiliary supervision significantly degrades primary MTL.

Table 4: **H3** Results. *: the model performance has a statistically significant difference with the baseline model.

	IEMOCAP				**SEMAINE**			
Hierarchy	**Classif. MA(4)**	**Val. MAE**	**Act. MAE**	**Dom. MAE**	**Act. MAE**	**Int. MAE**	**Pow. MAE**	**Val. MAE**
Flat (Baseline)	0.699	0.520	0.526	0.606	0.183	0.230	0.164	0.185
HAN-ROCK	0.715	0.538	0.507*	0.600*	0.184	0.218*	0.167*	0.178*

6 CONCLUSION

We proposed to re-use feature engineering pre-processing data as auxiliary tasks in primary MTL. Three hypotheses were tested for primary MTL. The experimental results confirm **H1** – Introducing auxiliary supervision resulted in better performance in most primary tasks. For **H2**, maximum learning capacity should be given to the primary branch. Lastly, for **H3**, placing the auxiliary supervision in a lower hierarchy is unlikely to hurt performance significantly, and it sometimes significantly improves performance. This is good news for multi-modal conversational analysis systems as we have demonstrated how pre-processed data can be used twice to improve performance, once as inputs, and again as auxiliary tasks. A limitation here is the apriori selection of the auxiliary targets. Future research could investigate targets selection, including how to use a much larger range of auxiliary targets, how to decide the optimum number of auxiliary targets, and if it is possible to perform these automatically in an algorithm.

[2]As opposed to, say, assigning 1 GRU to the primary talkturn coder and 256 GRU to each of the sixteen auxiliary talkturn encoder.

REFERENCES

Anonymous. Monah: Multimodal narratives for humans. *Under review at EACL2021*, 2021.

Udit Arora, William Scott Paka, and Tanmoy Chakraborty. Multitask learning for blackmarket tweet detection. In *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pp. 127–130, 2019.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Tadas Baltrušaitis, Peter Robinson, and Louis-Philippe Morency. Openface: an open source facial behavior analysis toolkit. In *Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on*, pp. 1–10. IEEE, 2016.

Peter Bell, Pawel Swietojanski, and Steve Renals. Multitask learning of context-dependent targets in deep neural network acoustic models. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(2):238–247, 2016.

James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *The Journal of Machine Learning Research*, 13(1):281–305, 2012.

Johannes Bjerva. Will my auxiliary tagging task help? estimating auxiliary tasks effectivity in multi-task learning. In *Proceedings of the 21st Nordic Conference on Computational Linguistics*, pp. 216–220, 2017.

Stefano B Blumberg, Ryutaro Tanno, Iasonas Kokkinos, and Daniel C Alexander. Deeper image quality transfer: Training low-memory neural networks for 3d images. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 118–125. Springer, 2018.

Carlos Busso, Murtaza Bulut, Chi-Chun Lee, Abe Kazemzadeh, Emily Mower, Samuel Kim, Jeannette N Chang, Sungbok Lee, and Shrikanth S Narayanan. IEMOCAP: Interactive emotional dyadic motion capture database. *Language resources and evaluation*, 42(4):335, 2008.

Rich Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.

Rich Caruana and Virginia R De Sa. Promoting poor features to supervisors: Some inputs work better as outputs. In *Advances in Neural Information Processing Systems*, pp. 389–395, 1997.

Rich Caruana and Virginia R de Sa. Using feature selection to find inputs that work better as extra outputs. In *International Conference on Artificial Neural Networks*, pp. 299–304. Springer, 1998.

Rich Caruana, Shumeet Baluja, and Tom Mitchell. Using the future to "sort out" the present: Rankprop and multitask learning for medical risk evaluation. In *Advances in neural information processing systems*, pp. 959–965, 1996.

Dongpeng Chen and Brian Kan-Wing Mak. Multitask learning of deep neural networks for low-resource speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(7):1172–1183, 2015.

Dongpeng Chen, Brian Mak, Cheung-Chi Leung, and Sunil Sivadas. Joint acoustic modeling of triphones and trigraphemes by multi-task learning deep neural networks for low-resource speech recognition. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5592–5596. IEEE, 2014.

Hao Cheng, Hao Fang, and Mari Ostendorf. Open-domain name error detection using a multi-task rnn. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 737–746, 2015.

Maximilian Christ, Nils Braun, Julius Neuffer, and Andreas W Kempa-Liehr. Time series feature extraction on basis of scalable hypothesis tests (tsfresh–a python package). *Neurocomputing*, 307: 72–77, 2018.

Yongping Du, Yunpeng Pan, and Junzhong Ji. A novel serial deep multi-task learning model for large scale biomedical semantic indexing. In *2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pp. 533–537. IEEE, 2017.

Rosenberg Ekman. *What the face reveals: Basic and applied studies of spontaneous expression using the Facial Action Coding System (FACS)*. Oxford University Press, USA, 1997.

Anna Fariha. Automatic image captioning using multitask learning. In *In the Proceedings of Neural Information Processing Systems*, volume 20, pp. 11–20, 2016.

Jose Maria Garcia-Garcia, Victor MR Penichet, and Maria D Lozano. Emotion detection: a technology review. In *Proceedings of the XVIII international conference on human computer interaction*, pp. 1–8, 2017.

Daniel Golovin, Benjamin Solnik, Subhodeep Moitra, Greg Kochanski, John Karro, and D Sculley. Google vizier: A service for black-box optimization. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1487–1495, 2017.

Ting Gong, Tyler Lee, Cory Stephenson, Venkata Renduchintala, Suchismita Padhy, Anthony Ndirango, Gokce Keskin, and Oguz H Elibol. A comparison of loss weighting strategies for multi task learning in deep neural networks. *IEEE Access*, 7:141627–141632, 2019.

Judith A Hall, Debra L Roter, Danielle C Blanch, and Richard M Frankel. Observer-rated rapport in interactions between medical students and standardized patients. *Patient Education and Counseling*, 76(3):323–327, 2009.

Kaveh Hassani and Mike Haley. Unsupervised multi-task feature learning on point clouds. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 8160–8171, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Mohammed Hoque, Matthieu Courgeon, Jean-Claude Martin, Bilge Mutlu, and Rosalind W Picard. Mach: My automated conversation coach. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pp. 697–706, 2013.

Yanping Huang, Youlong Cheng, Ankur Bapna, Orhan Firat, Dehao Chen, Mia Chen, HyoukJoong Lee, Jiquan Ngiam, Quoc V Le, Yonghui Wu, et al. Gpipe: Efficient training of giant neural networks using pipeline parallelism. In *Advances in neural information processing systems*, pp. 103–112, 2019.

Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An introduction to statistical learning*, volume 112. Springer, 2013.

Joshua Y Kim, Rafael A Calvo, Kalina Yacef, and NJ Enfield. A review on dyadic conversation visualizations-purposes, data, lens of analysis. *arXiv preprint arXiv:1905.00653*, 2019a.

Joshua Y Kim, Greyson Y Kim, and Kalina Yacef. Detecting depression in dyadic conversations with multimodal narratives and visualizations. In *Australasian Joint Conference on Artificial Intelligence*, pp. 303–314. Springer, 2019b.

Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Physical review E*, 69(6):066138, 2004.

Kalpesh Krishna, Shubham Toshniwal, and Karen Livescu. Hierarchical multitask learning for ctc-based speech recognition. *arXiv preprint arXiv:1807.06234*, 2018.

Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

Giwoong Lee, Eunho Yang, and Sung Hwang. Asymmetric multi-task learning based on task relatedness and loss. In *International Conference on Machine Learning*, pp. 230–238, 2016.

Hae Beom Lee, Eunho Yang, and Sung Ju Hwang. Deep asymmetric multi-task feature learning. In *International Conference on Machine Learning*, pp. 2956–2964. PMLR, 2018.

Shengchao Liu, Yingyu Liang, and Anthony Gitter. Loss-balanced task weighting to reduce negative transfer in multi-task learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 9977–9978, 2019.

Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pp. 21–37. Springer, 2016.

Navonil Majumder, Soujanya Poria, Devamanyu Hazarika, Rada Mihalcea, Alexander Gelbukh, and Erik Cambria. Dialoguernn: An attentive rnn for emotion detection in conversations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 6818–6825, 2019.

Gary McKeown, Michel Valstar, Roddy Cowie, Maja Pantic, and Marc Schroder. The semaine database: Annotated multimodal records of emotionally colored conversations between a person and a limited agent. *IEEE transactions on affective computing*, 3(1):5–17, 2011.

Afonso Menegola, Michel Fornaciali, Ramon Pires, Flávia Vasques Bittencourt, Sandra Avila, and Eduardo Valle. Knowledge transfer for melanoma screening with deep learning. In *2017 IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017)*, pp. 297–300. IEEE, 2017.

Taylor Mordan, Nicolas Thome, Gilles Henaff, and Matthieu Cord. Revisiting multi-task learning with rock: a deep residual auxiliary block for visual detection. In *Advances in Neural Information Processing Systems*, pp. 1310–1322, 2018.

Iftekhar Naim, M Iftekhar Tanveer, Daniel Gildea, and Mohammed Ehsan Hoque. Automated prediction and analysis of job interview performance: The role of what you say and how you say it. In *2015 11th IEEE international conference and workshops on automatic face and gesture recognition (FG)*, volume 1, pp. 1–6. IEEE, 2015.

Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. Deep double descent: Where bigger models and more data hurt. *arXiv preprint arXiv:1912.02292*, 2019.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014. ISBN 9781937284961.

Soujanya Poria, Navonil Majumder, Rada Mihalcea, and Eduard Hovy. Emotion recognition in conversation: Research challenges, datasets, and recent advances. *IEEE Access*, 7:100943–100953, 2019.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*, 2014.

Brian C Ross. Mutual information between discrete and continuous data sets. *PloS one*, 9(2): e87357, 2014.

Najmeh Sadoughi and Carlos Busso. Expressive speech-driven lip movements with multitask learning. In *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, pp. 409–415. IEEE, 2018.

Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In *Advances in Neural Information Processing Systems*, pp. 527–538, 2018.

Jaak Simm, Ildefons Magrans de Abril, and Masashi Sugiyama. Tree-based ensemble multi-task learning method for classification and regression. *IEICE TRANSACTIONS on Information and Systems*, 97(6):1677–1681, 2014.

M Iftekhar Tanveer, Emy Lin, and Mohammed Hoque. Rhema: A real-time in-situ intelligent interface to help people with public speaking. In *Proceedings of the 20th International Conference on Intelligent User Interfaces*, pp. 286–295, 2015.

Fei Tao and Carlos Busso. End-to-end audiovisual speech recognition system with multitask learning. *IEEE Transactions on Multimedia*, 2020.

Lisa Torrey and Jude Shavlik. Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, pp. 242–264. IGI global, 2010.

Vokaturi. Vokaturi Overview, 2019. URL `https://developers.vokaturi.com/getting-started/overview`.

Sen Wu, Hongyang R Zhang, and Christopher Ré. Understanding and improving information transfer in multi-task learning. *arXiv preprint arXiv:2005.00944*, 2020.

Jianliang Yang, Yuenan Liu, Minghui Qian, Chenghua Guan, and Xiangfei Yuan. Information extraction from electronic medical records using multitask recurrent neural network with contextual word embedding. *Applied Sciences*, 9(18):3658, 2019.

Le Yang, Dongmei Jiang, Lang He, Ercheng Pei, Meshia Cédric Oveneke, and Hichem Sahli. Decision tree based depression classification from audio video and language information. In *Proceedings of the 6th international workshop on audio/visual emotion challenge*, pp. 89–96, 2016a.

Min Yang, Wei Zhao, Wei Xu, Yabing Feng, Zhou Zhao, Xiaojun Chen, and Kai Lei. Multitask learning for cross-domain image captioning. *IEEE Transactions on Multimedia*, 21(4):1047–1061, 2018.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1480–1489, 2016b.

ByungIn Yoo, Youngjun Kwak, Youngsung Kim, Changkyu Choi, and Junmo Kim. Deep facial age estimation using conditional multitask learning with weak label expansion. *IEEE Signal Processing Letters*, 25(6):808–812, 2018.

Abdallah Yousif, Zhendong Niu, and Ally S Nyamawe. Citation classification using multitask convolutional neural network model. In *International Conference on Knowledge Science, Engineering and Management*, pp. 232–243. Springer, 2018.

Jianfei Yu and Jing Jiang. Learning sentence embeddings with auxiliary tasks for cross-domain sentiment classification. Association for Computational Linguistics, 2016.

Nasser Zalmout and Nizar Habash. Adversarial multitask learning for joint multi-feature and multi-dialect morphological modeling. *arXiv preprint arXiv:1910.12702*, 2019.

A APPENDIX

A.1 DATASET PARTITIONS

We detail the dataset partition in 5 for aid reproducibility.

Table 5: Dataset partitions

IEMOCAP			SEMAINE		
Partition	**Session**	**Count**	**Partition**	**Session**	**Count**
Train	Ses01F	861	Train	2008.12.05.16.03.15	424
Train	Ses01M	958	Train	2009.01.30.12.00.35	404
Train	Ses02F	889	Train	2009.02.12.10.49.45	686
Train	Ses02M	922	Train	2009.05.15.15.04.29	436
Train	Ses03F	958	Train	2009.05.29.14.30.05	760
Train	Ses03M	1178	Train	2009.05.22.15.17.45	668
Dev	Ses04F	1105	Train	2009.05.25.11.23.09	928
Dev	Ses04M	998	Train	2009.05.26.10.19.53	790
Test	Ses05F	1128	Train	2009.06.05.10.14.28	732
Test	Ses05M	1042	Train	2009.06.15.12.13.06	958
			Train	2009.06.19.14.01.24	586
			Train	2009.10.27.16.17.38	440
			Dev	2008.12.19.11.03.11	188
			Dev	2009.01.06.14.53.49	472
			Dev	2009.05.12.15.02.01	448
			Dev	2009.05.08.11.28.48	752
			Dev	2009.06.26.14.38.17	404
			Test	2008.12.14.14.47.07	372
			Test	2009.01.06.12.41.42	264
			Test	2009.01.28.15.35.20	364
			Test	2009.06.26.14.38.17	440
			Test	2009.06.26.14.09.45	448

A.2 SCALING THE AUXILIARY TARGETS

We detail the operations in scaling the percentile scores that range [0,1] to various primary tasks. For IEMOCAP Primary Tasks that are regression problems, we multiply the percentile score by 4 and add 1 to obtain the range [1,5]. For IEMOCAP classification task, we leave the auxiliary targets in the range of [0,1]. As for SEMAINE tasks, which are all regression problems, we multiply the percentile score by 2 and minus 1 to obtain the range [-1,1].

A.3 RANGE OF HYPER PARAMETERS TUNED

We document the range of hyper parameters tuned in Table 6 to aid reproducibility.

A.4 HYPER PARAMETERS WITH BEST DEVELOPMENT SET PERFORMANCE

We document our best performing hyper parameters in Table 7 to aid reproducibility.

A.5 DETAILS OF COMPUTING THE BOOTSTRAP CONFIDENCE INTERVAL

Baseline models for each hypothesis are detailed in section 2. All non-baseline models are referred to as challenger models. We created 1000 bootstrap samples of the test set performance by (1) resampling the development set performance, then (2) selecting the set of hyperparameters that resulted in the best development set performance, and (3) looking up the test set performance given the set of best-performing hyperparameters for the development set. To judge whether the challenger outperforms the baseline, we computed the 95 percent confidence interval by (1) performing

Table 6: Range of Hyperparameters tuned. U: Uniform sampling, LU: Uniform sampling on the logarithmic scale.

Name	**Min.**	**Max**	**Sampling**
Learning rate	2^{-10}	2^{-5}	LU
Batch-Size	32	256	U
Pri GRU	1, 64, 128, 192, 256		U
Aux GRU	257 - (minus) Pri GRU		
Loss-weights Assignment	Random, Linear-Normalized, Softmax-Normalized		U
L2 Regularization	0.0	0.50	LU
Content-Size	1	30	U
GRU dropout	0.01	0.50	U
Recurrent dropout	0.01	0.50	U

element-wise subtraction between the resampled test set performance of the baseline against the challenger, (2) removing the top and bottom 2.5 percent from the differences, and (3) observing whether the remaining 95 percent confidence interval includes zero. If it does not include zero, then the difference is statistically significant.

Table 7: Best Hyperparameters settings. R: Random, L: Linear-Normalized, S: Softmax-Normalized.

	IEMOCAP				**SEMAINE**			
	Classif. MA(4)	**Val. MAE**	**Act. MAE**	**Dom. MAE**	**Act. MAE**	**Int. MAE**	**Pow. MAE**	**Val. MAE**
Dev. set	0.714	0.482	0.492	0.570	0.118	0.164	0.135	0.138
Test set	0.747	0.497	0.499	0.587	0.184	0.215	0.164	0.169
Learning rate	6.21 e-03	2.21 e-02	1.44 e-02	3.13 e-02	2.43 e-02	1.58 e-02	3.04 e-03	2.94 e-02
Batch size	43	43	62	77	41	43	33	62
Pri GRU	256	256	256	256	256	256	192	256
Aux GRU	1	1	1	1	1	1	65	1
L2 regular-ization	2.25 e-05	0	1.43 e-05	2.41 e-04	0	0	0	0
Content-Size	18	4	3	3	17	21	19	14
GRU dropout	0.27	0.49	0.33	0.30	0.06	0.17	0.05	0.07
Recurrent dropout	0.04	0.03	0.02	0.32	0.25	0.28	0.08	0.26
Epoch No.	232	161	74	14	48	116	110	110
Aux. weights assignment	S	R	L	L	L	R	S	R
Main loss weight	0.840	0.560	0.810	0.610	0.870	0.940	0.950	0.920
AU05 loss weight	0.009	0.013	0.014	0.000	0.000	0.002	0.002	0.008
AU17 loss weight	0.009	0.027	0.009	0.003	0.002	0.004	0.002	0.003
AU20 loss weight	0.009	0.028	0.0	0.001	0.000	0.007	0.002	0.006
AU25 loss weight	0.009	0.031	0.006	0.002	0.015	0.008	0.002	0.009
Happy tone loss weight	0.010	0.023	0.038	0.006	0.021	0.013	0.002	0.003
Sad tone loss weight	0.010	0.028	0.053	0.008	0.037	0.010	0.002	0.003
Angry tone loss weight	0.010	0.016	0.034	0.006	0.055	0.011	0.002	0.000
Fear tone loss weight	0.010	0.043	0.036	0.004	0.000	0.004	0.002	0.003
Y_{t-1} loss weight	0.010	0.030	0.0	0.040	0.000	0.000	0.005	0.006
Y_{t-2} loss weight	0.011	0.022	0.0	0.049	0.000	0.000	0.004	0.004
Y_{t-3} loss weight	0.010	0.006	0.0	0.046	0.000	0.000	0.003	0.006
Y_{t-4} loss weight	0.010	0.040	0.0	0.044	0.000	0.000	0.003	0.007
Y_{t+1} loss weight	0.010	0.042	0.0	0.042	0.000	0.000	0.005	0.005
Y_{t+2} loss weight	0.011	0.0	0.0	0.052	0.000	0.000	0.004	0.006
Y_{t+3} loss weight	0.010	0.050	0.0	0.043	0.000	0.000	0.003	0.008
Y_{t+4} loss weight	0.010	0.038	0.0	0.044	0.000	0.000	0.003	0.003