XFlow: 1D ↭ 2D Cross-modal Deep Neural Networks for Audiovisual Classification

Cătălina Cangea[1], Petar Veličković[1] and Pietro Liò[1]

***Abstract*— We propose two *multimodal deep learning architectures* that allow for cross-modal dataflow (XFlow) between the feature extractors, thereby extracting more interpretable features and obtaining a better representation than through unimodal learning, for the same amount of training data. These models can usefully exploit correlations between audio and visual data, which have a different dimensionality and are therefore *nontrivially exchangeable*. Our work improves on existing multimodal deep learning metholodogies in two essential ways: (1) it presents a novel method for performing cross-modality (*before* features are learned from individual modalities) and (2) extends the previously proposed *cross-connections* [1], which only transfer information between streams that process *compatible* data. Both cross-modal architectures outperformed their baselines (by up to 7.5%) when evaluated on the *AVletters* dataset.**

I. INTRODUCTION

An interesting extension of unimodal learning consists of deep models which "fuse" several modalities (for example, sound, image or text) and thereby learn a shared representation, outperforming previous architectures on discriminative tasks. However, the cross-modality in existing models only occurs after the features are learned [2], [3], [4], thereby preventing the unimodal feature extractors from exploiting any information contained within the other modalities. The work presented in this paper has focused on enabling information about the modalities to be exchanged while extracting more interpretable features, making it possible to exploit the correlations between several types of data more directly.

This information exchange may occur between data of varying dimensionality (for example, 1D/2D for audiovisual data) and thus poses a highly nontrivial problem. The idea behind this method is to use the correlations between different kinds of data to learn a better representation than the one which would result from combining independent unimodal feature extractors, given the same amount of training data.

II. CROSS-MODALITY FOR AUDIOVISUAL DATA

A. CNN × MLP

Illustrated in Figure 1, the first multimodal architecture takes as input a tuple `(x_img, x_mfcc)` and outputs a probability distribution over the possible classes that this example belongs to. The first element represents a 2D visual modality (the video frames for a person saying a letter) and is processed by a convolutional neural network, whereas the second one consists of 1D audio data corresponding to the same frames, in the form of mel-frequency cepstral coefficients (MFCCs), and is fed into a multi-layer perceptron.

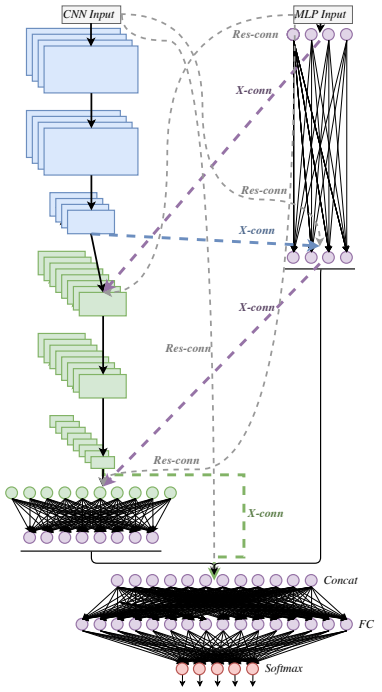

Fig. 1: CNN × MLP model with cross- and residual connections. The former are shown in blue and green, respectively, while the latter are illustrated in grey.

This architecture can only process fixed-size inputs, so we had to perform averaging over video frames and corresponding MFCC sets for all examples in the dataset. Since the length of a video in an example can vary considerably from person to person, some examples had to undergo averaging over a large window size. This results in loss of information about the changes between consecutive frames, so we expected that this would hurt the performance of the model. Indeed, the architecture described in Subsection II-B, which processes variable-length examples, is capable of more accurate predictions.

The primary role of *cross-connections* is to perform information exchange while the features from individual modalities are being learned (that is, before the concatenation operation). A fundamental incompatibility exists between 1D and 2D data—there is no trivially interpretable way of transferring the feature maps resulting from a {conv×2,

[1]Computer Laboratory, University of Cambridge, United Kingdom {ccc53,pv273,pl219}@cam.ac.uk

max-pool} block to the fully-connected layers processing the audio data and vice versa. We therefore had to design more complex types of cross-connections that would enable the data to be exchanged in a sensible manner and allow useful interpretations of these transfers.

The 2D↝1D cross-connections (shown in blue and green, respectively, in Figure 1) have the following structure: the output of a {conv×2, max-pool} block in the CNN is passed through a convolutional layer. The result is then flattened and processed by a fully-connected layer. Finally, we concatenate the output of the latter with the output of the corresponding fully-connected layer in the MLP (for the first cross-connection) or directly with the outputs of the CNN and MLP (for the second one). The 1D↝2D cross-connections (shown in purple) perform the inverse operation: the output of a fully-connected layer is passed through another layer of the same type, such that the number of features matches the dimensionality required for the *deconvolution* operation. We apply the latter to the reshaped data and concatenate the result with the output of the corresponding {conv×2, max-pool} block.

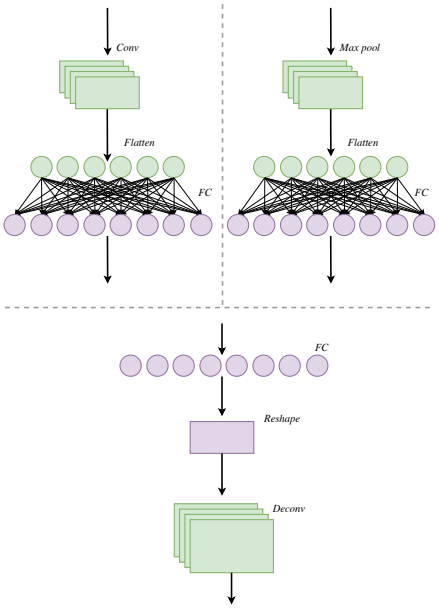

Fig. 2: Detailed view of the connections. *(Upper left:)* 2D↝1D cross-connection. *(Upper right:)* 2D↝1D residual connection. *(Bottom:)* 1D↝2D cross-/residual connection.

Residual learning [5] has the purpose of making the internal layers in a neural network represent the data more accurately. Our cross-connection design allows for straightforwardly including residual cross-modal connections that allow the raw input of one modality to directly interact with another modality's intermediate representations. This effectively has the potential to correct for any unwanted effects that one stream's intermediate transformations might have caused. Figure 2 also illustrates residual connections, constructed in a similar manner to cross-connections.

Mathematically, a 1D↝2D cross-connection with residuals will add $\mathit{Reshape}(\mathbf{W}_{\text{res}}\mathbf{x}_{\text{in}} + \mathbf{b}_{\text{res}}) * \mathbf{K}_{\text{res}}$ to the 2D stream at depth d and concatenate $\mathit{Reshape}(\mathbf{W}_{\text{xcon}}\mathbf{h}_d + \mathbf{b}_{\text{xcon}}) * \mathbf{K}_{\text{xcon}}$, where $\mathbf{W}_*$ and $\mathbf{K}_*$ are learnable weights, $\mathbf{b}_*$ are learnable biases, $\mathbf{x}_{\text{in}}$ are inputs and $\mathbf{h}_d$ are intermediate layer outputs.

B. {*CNN* × *MLP*}–*LSTM*

The second architecture processes the same kind of data as the CNN × MLP model, namely tuples of the form `(x_img, x_mfcc)`. However, the fundamental difference lies in the fact that each video frame/MFCCs pair is being provided separately as input to the pre-concatenation streams. This brings forward the crucial advantage of not having to average the data across more frames, keeping the temporal structure intact and maintaining a richer source of features from both modalities.

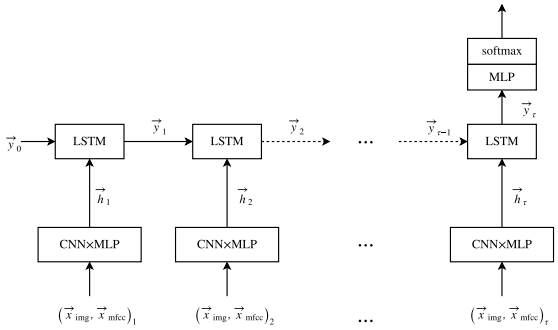

Fig. 3: {CNN × MLP}–LSTM macro-scale: sequential processing across time steps. The "CNN × MLP" rectangle represents the micro-scale per-frame extractor, shown in Figure 4. The two input modalities are denoted by $\vec{x}_{\text{img}}$ and $\vec{x}_{\text{mfcc}}$, while $\vec{y}_t$ is the output of the LSTM layer at time t.

Shown in Figure 4, the feature extractor for a single frame is *weight-shared* across all frames, which allows it to process input sequences of arbitrary lengths. After one set of features is extracted from the two modalities for each frame, it gets passed to an LSTM layer as an element $\vec{h}_i$ from the whole sequence, as illustrated in Figure 3. This layer then produces a set of 64 features for the entire example which is finally classified by the softmax layer. Additionally, all cross- and residual connections are designed in the same manner as for the CNN × MLP architecture, but differ in the sense that they only operate within the space of the single-frame feature extractor.

While similar to the CNN × MLP model, this one differs in that the second convolutional layer from each {conv×2, max-pool} block has been removed and the number of kernels from the remaining layers has been halved. The underlying motivation for this choice arises from the features no longer being extracted from an averaged block corresponding to an entire video, but rather from an individual frame, thereby heavily sparsifying the available information for a single input.

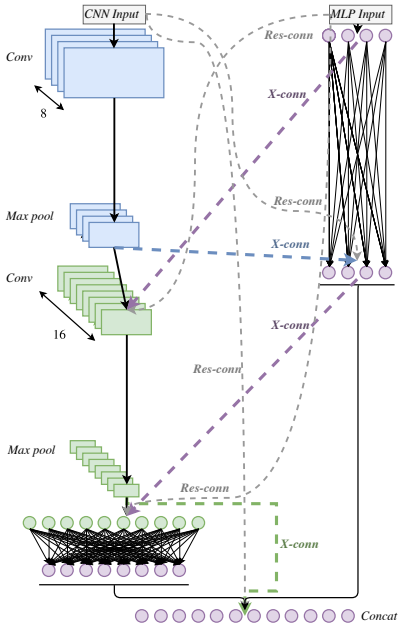

Fig. 4: CNN × MLP micro-scale: per-frame feature extractor with cross- and residual connections.

III. MODEL ARCHITECTURES

Tables I and II summarise the two models in terms of the number of parameters and cross-connections. For brevity, we have excluded the descriptions of residual connections, as they can be inferred from the shape of their target.

All convolutional and fully-connected layers in the architectures have *ReLU* activations. In terms of regularisation techniques for reducing overfitting in the CNN × MLP model, *batch normalisation* is applied after the input layer, each pair of convolutional layers, the first fully-connected layer in the MLP stream and the merge layer. We also applied *dropout* with $p = 0.25$ after every max-pooling layer and with $p = 0.5$ after the first fully-connected layer in the MLP stream, the merge layer and the final fully-connected layer. We chose a larger value for p in this case, due to the increased likelihood of overfitting in fully-connected layers, where the number of parameters is much larger than for convolutional layers. The {CNN × MLP}–LSTM model only employs batch normalisation after the input layer and merge layer, followed in the latter case by dropout with $p = 0.5$.

One important aspect is that an XFlow model is underregularised, compared to its baseline under the same regularisation parameters. Due to the increase in input size to the following layer, all merging points are passed through a dropout layer with $p = 0.5$. Additionally, when transmitting data across streams, we have taken steps to ensure integrity of the information. This is where the ReLU activation experiences a shortcoming—approximately half of its outputs are zero upon Xavier [6] initialisation. To enable the network to benefit from all transmitted data, we have applied the more general *PReLU* activation function [7], which allows for data to "leak" in the negative input space.

Output size	CNN stream	MLP stream
([80 × 60, 16], 128)	[3 × 3, 16] Conv × 2	Fully-connected 128-D
([40 × 30, 16], 128)	2 × 2 Max-Pool, stride 2	
([40 × 30, 32], 192)	[1 × 1, 16] Conv Fully-connected 64-D ↘	Fully-connected 759-D ↙ [8 × 8, 16] Deconv
([40 × 30, 32], 128)	[3 × 3, 32] Conv × 2	Fully-connected 128-D
([20 × 15, 32], 128)	2 × 2 Max-Pool, stride 2	
([20 × 15, 64], 256)	[1 × 1, 32] Conv Fully-connected 128-D ↘	Fully-connected 204-D ↙ [4 × 4, 32] Deconv
(256, 128)	Fully-connected 256-D	
512	Fully-connected 512-D	
	26-way softmax	

TABLE I: Architecture for the CNN × MLP baseline and model with cross-connections (whose parameters are described in blue).

Output size	CNN stream	MLP stream
([80 × 60, 8], 32)	[3 × 3, 8] Conv	Fully-connected 32-D
([40 × 30, 8], 32)	2 × 2 Max-Pool, stride 2	
([40 × 30, 8], 64)	[1 × 1, 8] Conv Fully-connected 32-D ↘	Fully-connected 375-D ↙ [16 × 16, 8] Deconv
([40 × 30, 16], 32)	[3 × 3, 16] Conv	Fully-connected 32-D
([20 × 15, 16], 32)	2 × 2 Max-Pool, stride 2	
([20 × 15, 64], 96)	[1 × 1, 16] Conv Fully-connected 64-D ↘	Fully-connected 104-D ↙ [8 × 8, 16] Deconv
(64, 32)	Fully-connected 64-D	
64	LSTM	
	26-way softmax	

TABLE II: Architecture for the {CNN × MLP}–LSTM baseline and model with cross-connections (whose parameters are described in blue).

IV. EVALUATION

We carried out a preliminary evaluation of the models using the *AVletters* dataset [8]. Spanning 26 classes representing the letters *A-Z*, *AVletters* contains 780 examples of 10 people saying each letter three times. The data was split into $k = 10$ folds to assess the performance of each classifier. Each of the folds corresponds to a *different person* in the dataset and can be seen as an extension to the usual *leave-one-out cross-validation* (LOOCV) approach, where each fold corresponds to *one* example. This allowed us to examine how well the models behave in a realistic audiovisual recognition setting—if we train a classifier with data collected from a group of people, we expect the model to be able to correctly identify the same information when being exposed to a new person.

Both architectures were trained using the Adam SGD optimiser for 200 epochs, with hyperparameters as described by Kingma and Ba [9] and a batch size of 128 for the CNN × MLP and 32 for the {CNN × MLP}–LSTM. The plots in Figures 5 and 6 show how the validation accuracy and cross-entropy loss, respectively, evolve as a function of the training epoch, for the {CNN × MLP}–LSTM model. A significant improvement over the baseline can be seen in both plots.

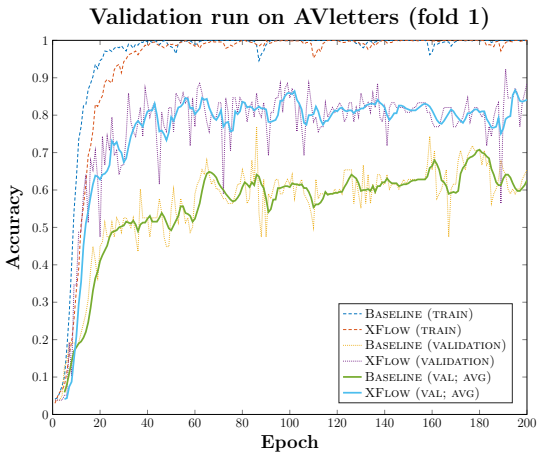

Fig. 5: Plot of the accuracy of the {CNN × MLP}–LSTM model on a single cross-validation fold. We have also used a sliding averaging window of 5 epochs on the accuracy values, to emphasise the model capabilities during training.

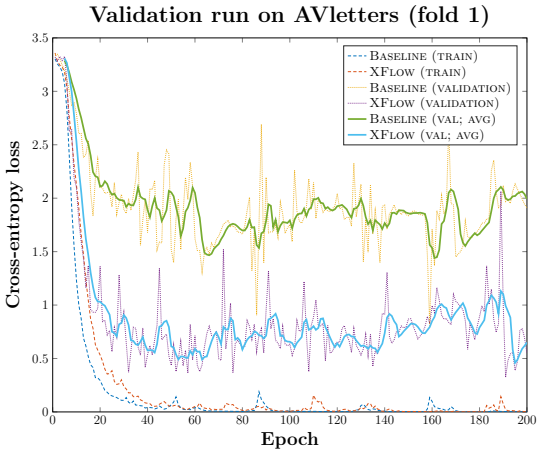

Fig. 6: Plot of the cross-entropy loss of the {CNN × MLP}–LSTM model on a single cross-validation fold.

As seen in Table III, the {CNN × MLP}–LSTM performed significantly better, with an improvement of 7.5% over its baseline. The resulting p-value was 0.02, which corresponds to a 98% confidence interval. However, the CNN × MLP only achieved a 0.9% edge over the model without cross-connections, its p-value not showing statistical significance. This illustrates the quality issues of *AVletters*—along with the averaging that results in loss of information, the discrepancy in visual and audio data pre-processing required us to average MFCC sets and video frames over time windows of different lengths, which likely resulted in misalignment of the visual and audio information.

V. CONCLUSIONS

Using newly developed cross- and residual connections to transform 1D to 2D representations and vice versa, we have designed two novel deep learning architectures for processing audiovisual data that could easily be applied to other kinds of information. Both residual and cross-connections were previously only used to process modalities that did not require intrinsic transformations. Consequently, the main challenge in building a new variety of cross-modal connections has lied in the fundamental incompatibility between the data types that are being exchanged. The novel cross-modality enabled both architectures to favourably exploit the correlations between modalities, outperforming their baselines on the *AVletters* dataset.

	Baseline	Cross-connected model	p-value
CNN × MLP	73.1%	**74.0%**	0.65
{CNN × MLP}–LSTM	78.1%	**85.6%**	<u>0.02</u>

TABLE III: Classification accuracy on the *AVletters* dataset for the two mentioned architectures. The p-values corresponding to statistically significant results are underlined here and in the following table.

The research we have presented in this paper could be extended to more modalities, potentially processing them in a hierarchical manner. Future directions also include investigating what representations are learned by the new cross-modal connections and the construction of a higher-quality dataset that would not suffer from over-processing and alignment issues.

REFERENCES

[1] P. Veličković, D. Wang, N. D. Lane, and P. Liò, "X-CNN: Cross-modal Convolutional Neural Networks for Sparse Datasets," *ArXiv e-prints*, Oct. 2016.

[2] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, "Multimodal Deep Learning," in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, 2011, pp. 689–696.

[3] N. Srivastava and R. R. Salakhutdinov, "Multimodal Learning with Deep Boltzmann machines," in *Advances in Neural Information Processing Systems*, 2012, pp. 2222–2230.

[4] Y. Aytar, C. Vondrick, and A. Torralba, "See, hear, and read: Deep aligned representations," *CoRR*, vol. abs/1706.00932, 2017. [Online]. Available: http://arxiv.org/abs/1706.00932

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[6] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks." in *Aistats*, vol. 9, 2010, pp. 249–256.

[7] K. He, X. Zhang, S. Ren, and J. Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[8] I. Matthews, T. Cootes, J. Bangham, S. Cox, and R. Harvey, "Extraction of visual features for lipreading," *IEEE Trans. on Pattern Analysis and Machine Vision*, vol. 24, no. 2, pp. 198–213, 2002.

[9] D. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv preprint arXiv:1412.6980*, 2014.