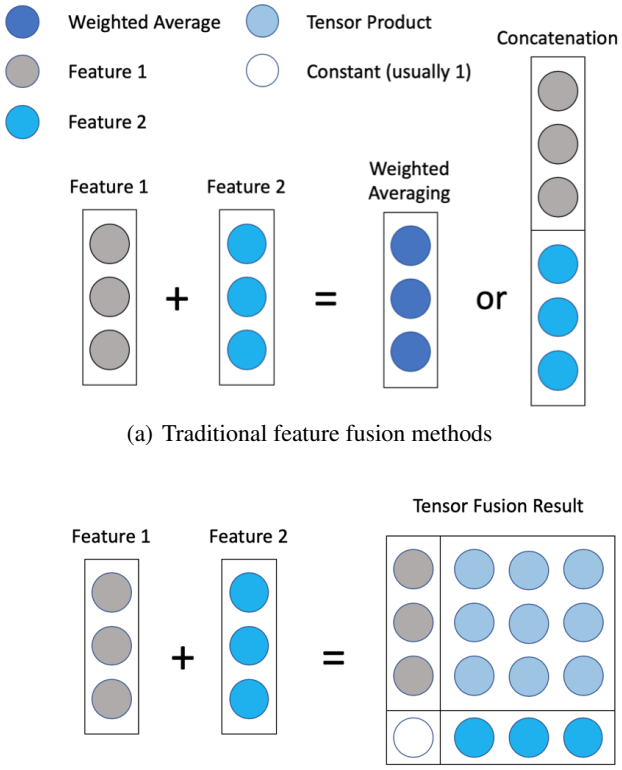
Generalized Tensor-based Multiple Feature Fusion Network using Block Decomposition

Paper ID: 2738

Abstract

Feature fusion based on neural networks has drawn great attention of researchers in the enhancement of system performance. Recent studies show the advantage of the tensor-based feature fusion methods, *e.g.*, Tensor Fusion Network, over the traditional early fusion and late fusion approaches given the superior power in the representation of complex dynamics. However, the computational complexity of the tensor-based feature fusion increases exponentially due to the tensor products. Some tensor decomposition methods are applied to solve the problem in previous work, such as Tucker decomposition and CP decomposition. In this paper, we propose a multiple feature fusion method using tensor Block decomposition. Theoretically, it is a general form of two previous tensor decomposition approaches and holds strengths from both of them. Based on the Block decomposition, we construct the corresponding Multiple feature Block Fusion Network (MBFN) practically. Our MBFN is applicable to both unimodal features and multimodal features and it can fuse multiple features at the same time. Experimental studies on a unimodal dataset (*i.e.*, Reuters) prove the effectiveness of multiple feature fusion. Experiments on two multimodal datasets (*i.e.*, MOSI and IEMOCAP) validate the generality and high-performance of the proposed MBFN.



(a) Traditional feature fusion methods

(b) Tensor-based feature fusion methods

Figure 1: Comparison between different feature fusion methods

1 Introduction

Deep learning research has shown numerous methods to solve problems in artificial intelligence. The ability for agents to combine information from multiple sources is valued because one source cannot provide enough information in many situations. Therefore, next-generation agents need to handle data from different domains with different models. Feature fusion methods in deep learning have drawn great attention of researchers. The application areas of feature fusion include face recognition [Hu *et al.*, 2017], video classification [Liu *et al.*, 2018; Zadeh *et al.*, 2017], and object recognition [Zhang *et al.*, 2018], *etc*.

Feature-level fusion methods can be divided into traditional feature aggregation methods and tensor-based subspace methods. Figure 1 shows the general view of these two broad categories. In traditional aggregation feature fusion methods, including concatenation and weighted averaging, the fused features are combinations of input features. Limited by the dimensions of the fused features, they cannot extract enough inter-information from original features. Tensor-based feature fusion methods have shown their strengths and advantages in many applications [Lei *et al.*, 2014; Zadeh *et al.*, 2017]. In these methods, the fused features contain not only the original features but also the tensor products of them.

Therefore, the fusion models have much richer representation of features and thus achieve better performance in different tasks.

Even though the existing tensor-based feature fusion methods are efficient in many problems, they cannot be used to fuse a large number of features due to the exponential computational complexity caused by tensor product. Tensor decomposition is an efficient method to solve the problem. It expresses a tensor as a sequence of elementary operations acting on other simpler tensors. Tucker decomposition [Tucker, 1966] and CANDECOMP/PARAFAC (CP) Decomposition [Carroll and Chang, 1970; Harshman, 1970] are two main tensor decomposition methods. Tucker decomposition decomposes a tensor into a core tensor multiplied (or transformed) by a matrix along each mode. G. Hu *et al*. [2017] used Tucker decomposition in the tensor product model to fuse two features. CP decomposition factorizes a tensor into a sum of components of rank-one tensors. The number of the rank-one tensors is the rank of this decomposition method. Liu *et al*. [2018] utilized low-rank CP decomposition in multiple feature fusion, which leverages low-rank weight tensors to fuse multimodal features effectively.

Although these two methods reduce the computational complexity of their tasks, both of the methods have their own weakness and limitations. The Tucker decomposition model is limited to fusing two features, while CP decomposition model cannot extract complex information in the original features. Additionally, the previous methods only focus on their specific problems and they lack versatility in feature fusion. To overcome these limitations, we firstly propose a multiple feature fusion method using tensor Block decomposition, a generalized tensor decomposition method combining both Tucker decomposition and CP decomposition [Kolda and Bader, 2009]. It is the first method to use tensor Block decomposition in feature fusion. The proposed feature fusion method holds strengths from both methods using Tucker decomposition and CP decomposition. To implement the feature fusion method in practice, we build the corresponding Multiple feature Block Fusion Network (MBFN), which is equivalent to the fusion method. This enables us to use the fusion method for concrete problems easily by building an end-to-end network. The proposed feature fusion method can fuse multiple features at the same time and the superiority of multiple feature fusion over two feature fusion is also verified. Our main contributions are twofold:

- We propose a novel multiple feature fusion method using tensor Block decomposition. It is the first method to use Block decomposition in feature fusion and holds both strengths from two previous tensor-based feature fusion methods on theoretical analysis. The multiple feature fusion architecture is more effective than the fusion of two features.
- We build a Multiple feature Block Fusion Network (MBFN) to implement the feature fusion method. It can be easily used in practical tasks as a unified and universal feature fusion method.

The rest of this paper is organized as follows. In the second section, we give a brief overview of related feature fusion and tensor decomposition methods. The third section presents our feature fusion methods using tensor Block decomposition and the corresponding network. In the fourth section, we give the experiment results on our networks and the analysis of them. The last section concludes the paper.

2 Related Work

Feature fusion. Feature fusion enables people to use data from different domains or extract features from one domain by different and complementary models. From the view of fusion position, feature-level (early fusion) and score-level (late fusion) methods are two main existing fusion methods. Some previous work made comparisons between feature-level and score-level fusion in pattern recognition [Gunes and Piccardi, 2005; Snoek *et al.*, 2005].

In score-level fusion, there are different models built with different input features. The final output of fusion is obtained from the outputs of these different models with rules. Some studies used fixed pre-defined fusion rules without any learnable parameters. For example, N. Thammasan *et al*. [2017] integrated multi-modal musical features together by weighted averaging on scores. K. Simonyan and A. Zisserman [2014] fused features from convolution neural networks for action recognition in videos by simple averaging. Some other studies used fusion models learned from data. Weighted averaging is used on scores for feature fusion where the weights are learned automatically in work [George and Routray, 2016; Ma *et al.*, 2015].

Feature-level fusion involves feature aggregation and subspace learning. In feature aggregation methods, features are fused by element-wise product or concatenation of vectors [Chen *et al.*, 2018]. In many subspace learning methods, features are first concatenated and then be projected into a new subspace. A. R. Chowdury *et al*. [2016] used BCNNs to fuse features from different layers in VGGNet and they demonstrated good results. The final features in the model are the same as applying polynomial kernel on the input features. The classical tensor-based feature fusion method is another subspace learning method to fuse features. The advantage of this method is that it takes the relationship between different features into consideration. B. Nojavanasghari *et al*. [2016] presented a deep multimodal fusion architecture to utilize complementary information from different modalities. Tensor Fusion Network (TFN) , a method using outer product of tensor is proposed to fuse multimodal features [Zadeh *et al.*, 2017]. The computational complexity of such model increases exponentially with the increase of features, resulting in its ineffectiveness in high dimensional tensor representations.

Feature fusion methods can also be divided into supervised and unsupervised fusion methods. The fusion methods based on simple concatenation, averaging, tensor product and Canonical Correlational Analysis (CCA) [Shawe-Taylor and Cristianini, 2004] are all unsupervised methods for the reason that they do not use any label information. On the contrary, supervised fusion methods, such as Linear Discriminant Analysis (LDA) [Belhumeur *et al.*, 1997] and Discriminant Correlation Analysis (DCA) [Haghighat *et al.*, 2016],

use label information and there are trainable parameters. Our feature fusion method is a supervised, feature-level, subspace learning method as it has trainable parameters and uses tensor products to fuse features.

Tensor decomposition. Tensor decomposition is an important approach in tensor-based feature fusion. It is usually used to solve the problem of computational complexity because it can express a tensor with simpler tensors of reduced sizes. Another advantage of tensor decomposition is that it can ease overfitting to some extent [Hu *et al.*, 2017]. Tucker decomposition and CP decomposition are two popular and classical tensor decomposition methods. G. Hu *et al.* [2017] applied Tucker decomposition in the fusion of face recognition features (FRF) and facial attribute features (FAF) to Gated Two-stream Neural Network (GTNN) in order to enhance face recognition performance. CP decomposition has been used to fuse unigram features, bigram features and trigram features for dependency parsing in [Lei *et al.*, 2014]. Another work [Liu *et al.*, 2018] is to use low-rank CP decomposition on fusing features from audio, video and text features, which makes multimodal fusion efficient without compromising on performance. Tensor Block decomposition [Kolda and Bader, 2009] is a general method which combines both CP decomposition and Tucker decomposition. Tucker decomposition and CP decomposition can be seen as two extreme cases of Block decomposition. Tensor Train decomposition is a relatively novel tensor decomposition method [Oseledets, 2011]. The power of feature fusion using tensor Train decomposition is presented in several previous papers [Khrulkov *et al.*, 2018; Novikov *et al.*, 2017].

3 Methodology

In this section, we present a multiple feature fusion method based on tensor Block decomposition and we compare it with other methods. A Multiple feature Block Fusion Network (MBFN) is further developed to implement the proposed feature fusion method.

3.1 Tensor-based Feature Fusion Method

Suppose we have n input feature vectors $\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_n$ with dimensions $A_1, A_2, \ldots, A_n$ and the size of the expected output feature $\mathbf{f}$ is C. The basic tensor-based fusion method is given by

$$\mathbf{f} = \mathcal{W} \times_1 \mathbf{f}_1 \times_2 \mathbf{f}_2 \times_3 \cdots \times_n \mathbf{f}_n. \tag{1}$$

The notation $\times_i$ is the tensor dot product (also known as tensor contraction), where i indicates at which axis the tensor dot product operates. $\mathcal{W} \in \mathbb{R}^{A_1 \times A_2 \times \cdots \times A_n \times C}$ is the fusion model parameter. Note that its size increases exponentially with the increase of the number of input features. Thus, we cannot use the fusion model directly for the sake of computation efficiency.

3.2 Multiple Feature Block Decomposition Method

Tensor approximation with tensor decomposition can decrease the computational complexity by reducing the number of the parameters in the fusion model. It can also accelerate the training speed and reduce overfitting [Hu *et al.*, 2017]. Motivated by the Block decomposition for tensors, we use a sum of low-rank Tucker tensors to approximate the fusion model parameter $\mathcal{W}$:

$$\mathcal{W} = \sum_{r=1}^{R} (\mathcal{S}^r \times_1 \mathbf{U}_1^r \times_2 \mathbf{U}_2^r \times_3 \cdots \times_n \mathbf{U}_n^r). \tag{2}$$

Here $\mathcal{S}^r \in \mathbb{R}^{a_1^r \times a_2^r \times \cdots \times a_n^r \times C}$ are low-rank Tucker tensors and $\mathbf{U}_i^r \in \mathbb{R}^{A_i \times a_i^r}$. Therefore, the number of parameters is reduced from $O(C\prod_{i=1}^{n} A_i)$ to $O(\sum_{r=1}^{R}(C\prod_{i=1}^{n} a_i^r + \sum_{i=1}^{n} a_i^r \times A_i))$, which grows linearly with the increase of the number of input features.

By subsituting Eq. (2) into Eq. (1), we have

$$\begin{aligned}\mathbf{f} = &\left(\sum_{r=1}^{R} (\mathcal{S}^r \times_1 \mathbf{U}_1^r \times_2 \mathbf{U}_2^r \times_3 \cdots \times_n \mathbf{U}_n^r)\right) \\ &\times_1 \mathbf{f}_1 \times_2 \mathbf{f}_2 \times_3 \cdots \times_n \mathbf{f}_n.\end{aligned} \tag{3}$$

Eq. (3) can be simplified as

$$\mathbf{f} = \sum_{r=1}^{R} (\mathcal{S}^r \times_1 \mathbf{U}_1^r \mathbf{f}_1 \times_2 \mathbf{U}_2^r \mathbf{f}_2 \times_3 \cdots \times_n \mathbf{U}_n^r \mathbf{f}_n). \tag{4}$$

Based on the property of Kronecker product, we can rewrite Eq. (4) as

$$\mathbf{f} = \sum_{r=1}^{R} \left((\mathcal{S}_{(C)}^r)^T (\mathbf{U}_1^r \mathbf{f}_1 \otimes \mathbf{U}_2^r \mathbf{f}_2 \otimes \cdots \otimes \mathbf{U}_n^r \mathbf{f}_n) \right). \tag{5}$$

Here $\otimes$ is the Kronecker product or outer product. $\mathcal{S}_{(C)}^r$ is the unfolding of $\mathcal{S}^r$ on the output mode as a matrix of size $\prod_{j=1}^{n} a_j^r \times C$. Its transpose $(\mathcal{S}_{(C)}^r)^T$ is a matrix of size $C \times \prod_{j=1}^{n} a_j^r$. The sum of the products of all $(\mathbf{U}_1^r \mathbf{f}_1 \otimes \mathbf{U}_2^r \mathbf{f}_2 \otimes \cdots \otimes \mathbf{U}_n^r \mathbf{f}_n) \in \mathbb{R}^{a_1 a_2 \ldots a_n \times C}$ and $(\mathcal{S}_{(C)}^r)^T \in \mathbb{R}^{1 \times a_1 a_2 \ldots a_n}$ is the fused feature in our Block decomposition method, a vector of dimension C. This fused feature covers the information from all original features.

3.3 Comparison of the Tensor Decomposition Methods

It is helpful to make a comparison between the feature fusion methods based on Tucker decomposition, CP decomposition and Block decomposition. We show the structures of the three tensor decomposition methods on a third-order tensor in Figure 2.

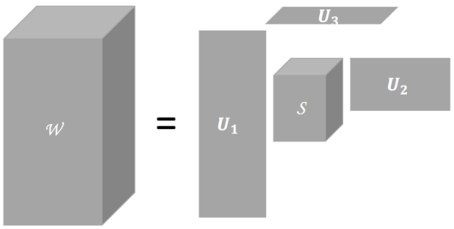
Existing methods using Tucker decomposition (*e.g.*, [Hu *et al.*, 2017]) can only fuse two features at the same time. To make a comparison with other methods, we generalize it into a multiple feature method:

$$\mathbf{f} = (\mathbf{U}_n \mathbf{f}_n \otimes \mathbf{U}_{n-1} \mathbf{f}_{n-1} \otimes \cdots \otimes \mathbf{U}_1 \mathbf{f}_1) \mathcal{S}_{(C)}^T. \tag{6}$$

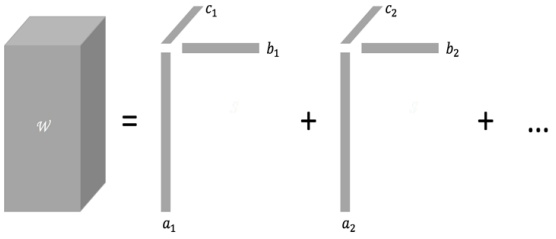
It is equivalent to a more complex model to extract information from features.

The fusion with CP decomposition [Liu *et al.*, 2018] can be modeled as follows:

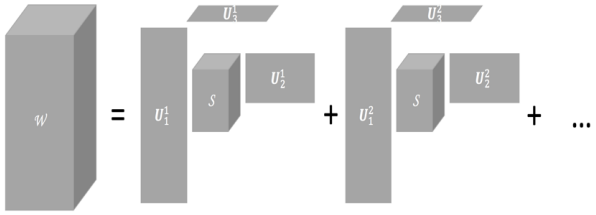
$$\mathbf{f} = \sum_{r=1}^{R} \left(\bigotimes_{i=1}^{n} \mathbf{w}_i^{(r)} \cdot \bigotimes_{i=1}^{n} \mathbf{f}_i \right). \tag{7}$$



(a) Tucker decomposition



(b) CP decomposition



(c) Block decomposition

Figure 2: The three tensor decomposition methods on a third-order tensor

It can be regarded as using R (rank) feature extractors to extract features separately. The mathematical expression of a single feature extractor is

$$\bigotimes_{i=1}^{n} \mathbf{w}_i^{(r)} \cdot \bigotimes_{i=1}^{n} \mathbf{f}_i. \tag{8}$$

The final fused feature is synthesized by all the intermediate results from the extractors, thus the model is general and the fused feature is more comprehensive. However, the complex information contained in original features is difficult to be extracted since each extractor is relatively simple.

The proposed Block decomposition method combines the above two methods. As shown in Eq. (5), it contains R relatively complex extractors like that in the Tucker decomposition method. It synthesizes multiple intermediate results as the final fused feature. Unlike the two feature Tucker decomposition method, our multiple feature fusion method using Block decomposition can fuse all features at the same time. Features are transmitted to the decision layer directly, thus the multiple feature fusion architectures are more effective than the others.

Note that the fusion methods using Tucker and CP decomposition are two extremes of Block decomposition. If we choose $R = 1$ or $a_1 = a_2 = \cdots = a_n = 1$ in the Block decomposition method, it will degenerate into the Tucker decomposition or CP decomposition fusion method. This fact proves that our Block decomposition method is more general, therefore it can be applied to broader situations.

3.4 Multiple feature Block Fusion Network (MBFN)

We present the neural network structure to implement the Block decomposition in practice. We use the same a_i^r for all rank r, so that we can use a three-order tensor $\mathbf{U}_i \in \mathbb{R}^{R \times A_i \times a_i}$ to represent all $\mathbf{U}_i^r$, and use an $(n+2)$-order tensor $\mathcal{S} \in \mathbb{R}^{R \times C \times a_1 \times a_2 \times \cdots \times a_n}$ to represent all $\mathcal{S}^r$. $\mathcal{S}^r$ is flattened into $\mathcal{S}_{(C)}^r \in \mathbb{R}^{\prod_{j=1}^{n} a_j^r \times C}$ as in Eq. (5), hence what we actually use is a matrix. Moreover, we use weighted summation rather than simple summation in the final step. The implementation of the step is to add a fully-connected layer at last. Thus the parameters in the neural network are $\mathbf{U}_i \in \mathbb{R}^{R \times A_i \times a_i} (i = 1, \ldots, n)$, $\mathcal{S} \in \mathbb{R}^{R \times C \times a_1 \times a_2 \times \cdots \times a_n}$, and $\mathbf{w} \in \mathbb{R}^R$. Therefore, the fusion method that we use practically is

$$\mathbf{f} = \mathbf{w}(\mathcal{S} \times (\mathbf{U}_1 \mathbf{f}_1 \otimes \mathbf{U}_2 \mathbf{f}_2 \otimes \cdots \otimes \mathbf{U}_n \mathbf{f}_n)). \tag{9}$$

The structure of the neural network to implement feature fusion using Block decomposition is shown in Figure 3. We denote this architecture as a Multiple feature Block Fusion Network (MBFN). The key layer in MBFN is the Kronecker

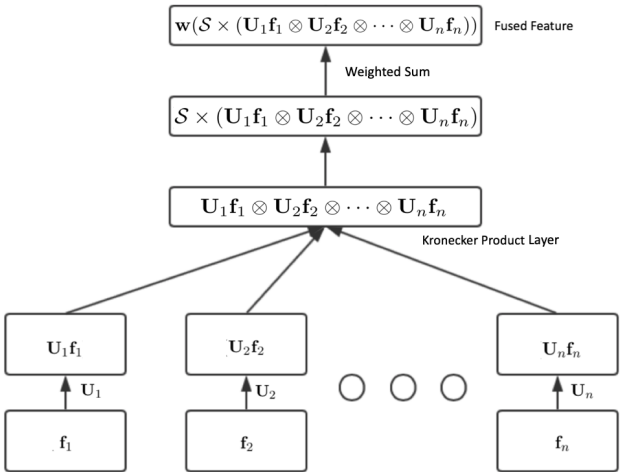


Figure 3: Network to implement feature fusion using Block decomposition

Product layer, which calculates the Kronecker Product $\mathbf{K}$ of input vectors. Given input vectors $\{\mathbf{U}_1 \mathbf{f}_1, \mathbf{U}_2 \mathbf{f}_2, \ldots, \mathbf{U}_n \mathbf{f}_n\}$, the output of a Kronecker Product layer is given by

$$\mathbf{K} = \mathbf{U}_1 \mathbf{f}_1 \otimes \mathbf{U}_2 \mathbf{f}_2 \otimes \cdots \otimes \mathbf{U}_n \mathbf{f}_n, \tag{10}$$

where $\mathbf{U}_i \mathbf{f}_i$ is a vector of dimension s_i and $\mathbf{K}$ is a vector of dimension $\prod_{j=1}^{n} s_j$. The elements in $\mathbf{K}$ are

$$K_i = (\mathbf{U}_1 \mathbf{f}_1)_{i_1} (\mathbf{U}_2 \mathbf{f}_2)_{i_2} \ldots (\mathbf{U}_n \mathbf{f}_n)_{i_n}, \tag{11}$$

where the relationship between i and $i_1, i_2, \dots i_n$ is

$$\begin{aligned} i &= i_1 s_2 s_3 \dots s_n + i_2 s_3 \dots s_n + \dots + i_{n-1} s_n + i_n \\ &= \sum_{k=1}^{n-1} \left(i_k \prod_{j=i+1}^{n} s_j \right) + i_n. \end{aligned} \tag{12}$$

The total number of parameters in MBFN is $R\left(\sum_{i=1}^{n}(A_i a_i) + (a_1 a_2 \dots a_n C)\right)$, which grows linearly with the increase of the input feature number. In testing and application, the fusion parameter $\mathcal{W}$ can be synthesised by the trainable parameters with Eq. (2).

4 Experiments

4.1 Effectiveness of Multiple Feature Fusion

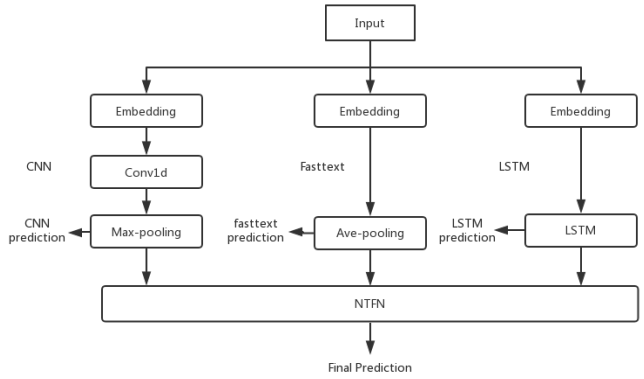
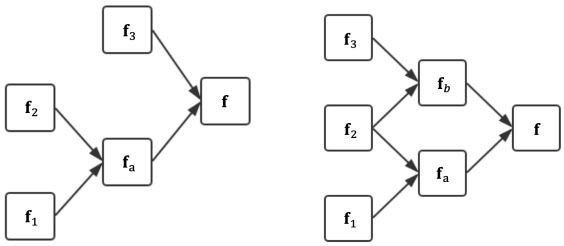


Figure 4: Neural network structure on Reuters Newswire Topics Classification dataset



(a) Tree-fusion structure (b) DAG-fusion structure

Figure 5: Tree-fusion structure and DAG-fusion structure for fusing three features using two-feature Tucker Network

To verify the effectiveness of our multiple feature fusion method, we carry out unimodal experiments on Reuters Newswire Topics Classification dataset. It is a text dataset with 11,228 newswires labeled over 46 topics. The task is to choose the corresponding topic for certain newswire, which is a multi-class classification problem. We use three methods to extract features from the original text. Fasttext [Grave *et al.*, 2017] is a simple but efficient method for text classification. CNN and LSTM are two classic methods to solve NLP problems. Here we maintain different models with different focuses. The CNN model concentrates on the local features of the text, while LSTM has a better understanding of global information. The structure of the whole network structure is shown in Figure 4. We use accuracy as the evaluation metric. We set the rank of MBFN to one to accelerate training and highlight the influence of multiple feature architecture. Also, the sizes of a_1, a_2 in Eq. (9) corresponding to $Fasttext$ and CNN models are relatively high in MBFN due to the better performance of these two single models.

Method		Accuracy (%)
Single Models	LSTM	68.9
	Fasttext	79.8
	CNN	80.3
Two-feature Tree-fusion	T_{LSTM}	79.4
	$T_{Fasttext}$	77.2
	T_{CNN}	80.0
Two-feature DAG-fusion	D_{LSTM}	77.0
	$D_{Fasttext}$	80.4
	D_{CNN}	78.0
Other Methods	Concatenation	80.4
	Average	80.5
MBFN		**81.2**

Table 1: Experiment Results on Reuters

We compare our MBFN with the two-feature fusion networks. A hierarchical fusion structure is applied to fuse three features with the two-feature fusion method. Two different fusion structures are shown in Figure 5. For the fusion method corresponding to the Tree structure shown in Figure 5 (a), we fuse two features firstly and then fuse the fused feature with the remaining feature. Hence, we have three different orders to fuse features by this fusion structure. In this section, we use $T_{\mathbf{f}_3}$ to represent the Tree-fusion model where feature $\mathbf{f}_3$ is fused in the end. For the fusion method corresponding to the Directed Acyclic Graph (DAG) structure shown in Figure 5 (b), a feature is firstly fused with the other two features respectively, then the two fused features are fused as the final fused feature. There are also three different fusion orders corresponding to the three original features with this structure. In this section, we use $D_{\mathbf{f}_2}$ to represent the DAG-fusion model where feature $\mathbf{f}_2$ is fused twice.

The experiment results are given in Table 1. The hierarchical two-feature fusion networks can only achieve the accuracy of 80.4% at most, and some of them are even worse than the single models. The performance of two simple feature fusion methods are slightly more than the two-feature fusion methods. Our multiple feature method MBFN can reach accuracy of 81.2%. This result proves that our generalization from two-feature method to multiple feature method is effective and necessary.

4.2 Comparison with State-of-the-art Methods

To compare our approach with state-of-the-art methods, we carry out experiments on two multimodal datasets: Mul-

timodal Opinion Sentiment Intensity (CMU-MOSI) dataset [Zadeh *et al.*, 2016] and Interactive Emotional Dyadic Motion Capture (IEMOCAP) [Busso *et al.*, 2008]. The data from both the datasets contains texts, videos and audios, thus we use the same networks to extract the three kinds of features. The whole network structure is shown in Figure 6. The models to extract the audio and video features are both three-layered fully-connected neural networks, and the model to extract the text feature is LSTM.

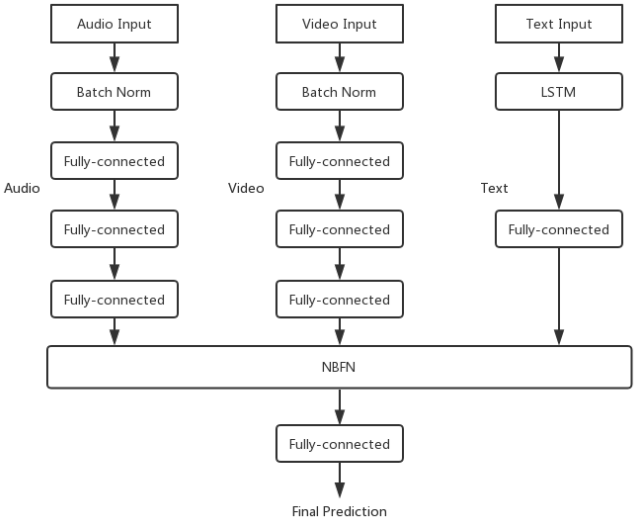


Figure 6: Neural network structure on multimodal datasets

CMU-MOSI CMU-MOSI is a dataset for multimodal sentiment intensity and subjectivity analysis. It contains 93 opinion videos from YouTube movie reviewers. Each video consists of segments, which are annotated with sentiments from -3 to 3. In our experiments, we regard the task as a binary classification problem. Our target is to judge whether the sentiment is positive or negative. The best hyperparameter settings are shown as follows. The sizes a_1, a_2 and a_3 in Eq. (9), which correspond to audio, video and text respectively in the MOSI dataset are 16, 8, and 64.

Method		Accuracy (%)	F1
Single Models	Audio Model	57.9	55.9
	Video Model	57.0	57.1
	Text Model	64.3	64.0
Feature Fusion Models	Concatenation	72.7	72.7
	Average	72.5	72.5
	ExM [2017]	74.1	73.7
	LMF [2018]	73.6	73.7
MBFN		**74.2**	**74.3**

Table 2: Experiment Results on MOSI

We compare our MBFN with two traditional feature fusion methods: Concatenation and Average, and two tensor-based feature fusion methods: ExM using tensor Train decomposition [Novikov *et al.*, 2017], LMF using low-rank CP decomposition [Liu *et al.*, 2018]. The experiment results are given in Table 2. The performance of the simple binary classifiers is poor because they only use the feature from one domain. The performance of tensor-based methods exceeds that of the traditional methods obviously. Our MBFN outperforms the other tensor-based feature fusion methods and works very well on this large and complex multi-modal dataset.

IEMOCAP IEMOCAP dataset is an emotion classification dataset of 302 videos. Our chosen task is to distinguish whether the speaker is happy. The sizes a_1, a_2 and a_3 in Eq. (9), which correspond to audio, video and text respectively in the MOSI dataset are 8, 8, and 64. Since the labels are unbalanced, we use F1-scores as the evaluation metric. We compare our MBFN with the three single models, two traditional feature fusion methods and LMF and other three methods: Deep Fusion [Nojavanasghari *et al.*, 2016], Tensor Fusion Network [Zadeh *et al.*, 2017] and LMF [Liu *et al.*, 2018].

The performance of different models is shown in Table 3. Due to the limitation of the dataset, tensor-based feature fusion methods do not show overwhelming advantage than traditional methods. However, LMF and our MBFN still outperform all the other methods, and our MBFN reaches the state-of-the-art performance on this dataset.

Method		F1
Single Models	Audio Model	79.1
	Video Model	83.3
	Text Model	79.6
Feature Fusion Models	Concatenation	85.5
	Average	84.3
	DF [2016]	81.0
	TFN [2017]	83.6
	LMF [2018]	**85.8**
MBFN		**85.8**

Table 3: Experiment Results on IEMOCAP

5 Conclusion

In this paper, we present a generalized tensor-based feature fusion method. Firstly, we propose a novel multiple feature fusion method using tensor Block decomposition, which is equivalent to constructing many powerful feature extractors in order to obtain better feature fusion results. It can fuse multiple features at the same time and is more effective than two feature fusion methods. Then, we build a Multiple feature Block Fusion Network (MBFN) to implement the proposed feature fusion method. Extensive experiments on unimodal and multimodal datasets show the effectiveness of the multiple feature fusion method and the better performance of the proposed framework than existing methods.

References

[Belhumeur *et al.*, 1997] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman. Eigenfaces vs. fisherfaces: recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):711–720, 1997.

[Busso *et al.*, 2008] Carlos Busso, Murtaza Bulut, Chi-Chun Lee, Abe Kazemzadeh, Emily Mower, Samuel Kim, Jean nette Chang, Sungbok Lee, and Shrikanth S. Narayanan. Iemocap: Interactive emotional dyadic motion capture database. *Journal of Language Resources and Evaluation*, 42(4):335–359, 2008.

[Carroll and Chang, 1970] J Douglas Carroll and Jih-Jie Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition. *Psychometrika*, 35(3):283–319, 1970.

[Chen *et al.*, 2018] Yilun Chen, Zhicheng Wang, Yuxiang Peng, Zhiqiang Zhang, Gang Yu, and Jian Sun. Cascaded pyramid network for multi-person pose estimation. In *CVPR*, 2018.

[Chowdhury *et al.*, 2016] Aruni Roy Chowdhury, Tsung Yu Lin, Subhransu Maji, and Erik Learned-Miller. One-to-many face recognition with bilinear cnns. In *WACV*, 2016.

[George and Routray, 2016] Anjith George and Aurobinda Routray. A score level fusion method for eye movement biometrics. *Pattern Recognition Letters*, 82:207 – 215, 2016.

[Grave *et al.*, 2017] Edouard Grave, Tomas Mikolov, Armand Joulin, and Piotr Bojanowski. Bag of tricks for efficient text classification. In *EACL*, 2017.

[Gunes and Piccardi, 2005] Hatice Gunes and Massimo Piccardi. Affect recognition from face and body: early fusion vs. late fusion. In *SMC*, 2005.

[Haghighat *et al.*, 2016] M. Haghighat, M. Abdel-Mottaleb, and W. Alhalabi. Discriminant correlation analysis: Real-time feature level fusion for multimodal biometric recognition. *IEEE Transactions on Information Forensics and Security*, 11:1984–1996, 2016.

[Harshman, 1970] R.A. Harshman. *Foundations of the PARAFAC Procedure: Models and Conditions for an "explanatory" Multi-modal Factor Analysis*. UCLA working papers in phonetics. 1970.

[Hu *et al.*, 2017] Guosheng Hu, Yang Hua, Yang Yuan, Zhihong Zhang, Zheng Lu, Sankha S. Mukherjee, Timothy M. Hospedales, Neil M. Robertson, and Yongxin Yang. Attribute-enhanced face recognition with neural tensor fusion networks. In *ICCV*, 2017.

[Khrulkov *et al.*, 2018] Valentin Khrulkov, Alexander Novikov, and Ivan Oseledets. Expressive power of recurrent neural networks. In *ICLR*, 2018.

[Kolda and Bader, 2009] T. Kolda and B. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, 2009.

[Lei *et al.*, 2014] Tao Lei, Yu Xin, Yuan Zhang, Regina Barzilay, and Tommi Jaakkola. Low-rank tensors for scoring dependency structures. In *ACL*, 2014.

[Liu *et al.*, 2018] Zhun Liu, Ying Shen, Varun Bharadhwaj Lakshminarasimhan, Paul Pu Liang, Amir Zadeh, and Louis-Philippe Morency. Efficient low-rank multimodal fusion with modality-specific factors. In *ACL*, 2018.

[Ma *et al.*, 2015] Lin Ma, Jiwen Lu, Jianjiang Feng, and Jie Zhou. Multiple feature fusion via weighted entropy for visual tracking. In *ICCV*, 2015.

[Nojavanasghari *et al.*, 2016] Behnaz Nojavanasghari, Deepak Gopinath, Jayanth Koushik, Tadas Baltrušaitis, and Louis-Philippe Morency. Deep multimodal fusion for persuasiveness prediction. In *ICMI*, 2016.

[Novikov *et al.*, 2017] Alexander Novikov, Mikhail Trofimov, and Ivan Oseledets. Exponential machines. In *ICLR*, 2017.

[Oseledets, 2011] Ivan V Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 2011.

[Shawe-Taylor and Cristianini, 2004] John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

[Simonyan and Zisserman, 2014] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, 2014.

[Snoek *et al.*, 2005] Cees GM Snoek, Marcel Worring, and Arnold WM Smeulders. Early versus late fusion in semantic video analysis. In *ACM MM*, 2005.

[Thammasan *et al.*, 2017] Nattapong Thammasan, Ken Ichi Fukui, and Masayuki Numao. Multimodal fusion of eeg and musical features in music-emotion recognition. In *AAAI*, 2017.

[Tucker, 1966] Ledyard R Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966.

[Zadeh *et al.*, 2016] Amir Zadeh, Rowan Zellers, Eli Pincus, and Louis-Philippe Morency. Mosi: Multimodal corpus of sentiment intensity and subjectivity analysis in online opinion videos. *arXiv preprint arXiv:1606.06259*, 2016.

[Zadeh *et al.*, 2017] Amir Zadeh, Minghai Chen, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. Tensor fusion network for multimodal sentiment analysis. In *EMNLP*, 2017.

[Zhang *et al.*, 2018] Xinyu Zhang, Hongbo Gao, Guopeng Li, Jianhui Zhao, Jianghao Huo, Jialun Yin, Yuchao Liu, and Li Zheng. Multi-view clustering based on graph-regularized nonnegative matrix factorization for object recognition. *Information Sciences*, 432:463 – 478, 2018.