SMARTHOME Product Technical Documentation

Zexin Fu, Weilin Ruan and Songning Lai

***Abstract*—This technical documentation serves as a comprehensive guide for developers and users interested in the SMARTHOME system. Chapter I provides a detailed overview of the SMARTHOME 1+N technology solution, which is a whole-house smart system designed to provide care services for the elderly. Chapter II discusses 3 pain points the system addresses. Chapter III introduces various smart devices that integrate audio sensors, cameras, and intelligent services such as health monitoring, multimodal sentiment analysis, and voice assistants. Chapter IV presents our multimodal sentiment analysis system. In Chapter V, we focus on enhancing the explainability and interpretability of our system, critical for user trust and acceptance. Finally, Chapter VI introduces FPGA based hardware acceleration for AI algorithms.**

I. Product Technology Framework

This whole-house smart system offers a 1+N technology solution. It utilizes the smart TV in the living room as the central hub for AI computing, device connectivity, and human-machine interaction. It integrates with N smart sensor products to provide care services for the physical and mental well-being of the elderly. The overall framework is depicted in Figure 1. The core technologies of the system include:

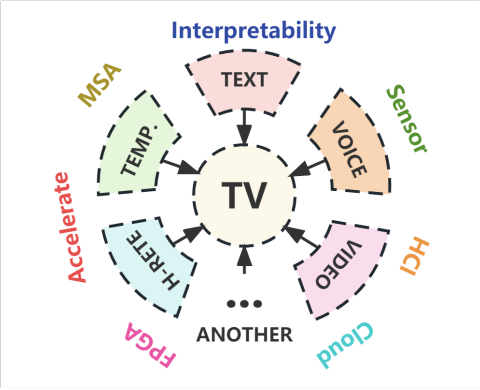

Fig. 1. 1+N Technology Framework

- Device Interconnectivity: The smart TV hub can connect up to 128 smart sensor products, receiving multimodal data from various sensing systems, including images, videos, audio, body temperature, heart rate, and more. The connectivity options support wireless WiFi and wired broadband connections, providing a flexible and high-speed IoT interconnection experience.
- AI Computing: The system focuses on the physical and mental health of the elderly and supports intelligent AI recognition functions such as fall detection, distress call detection, abnormal heart rate and body temperature alerts, and negative emotion detection. The system utilizes FPGA-based deep learning acceleration chips and CPUs to provide sufficient computing power for smart TVs and edge devices, ensuring efficient operation of the entire system.
- Multimodal Analysis: This system fully utilizes the various types of information provided by smart sensing devices to effectively perceive the health status and needs of elderly users. Unlike other products on the market that rely solely on single sensor data to identify physical health status, our system goes beyond that by integrating and analyzing multiple sensor data types such as video, audio, body temperature, heart rate, etc., to understand the user's psychological state. This allows us to provide comprehensive care that covers both the physical and mental aspects. Additionally, our system has interpretability, enabling it to provide more detailed emotional insights for a more personalized service experience.
- Human-Machine Interaction: To address the issue of elderly individuals being less proficient in using smart products, this system adopts a highly user-friendly proactive interaction approach. By detecting anomalies in the multimodal information from the diverse sensors, the system proactively offers caring greetings to the elderly and provides corresponding services such as engaging in conversation, offering medical assistance, and connecting with family members. This approach aims to enhance the user experience and ensure that the system actively supports and assists the elderly population.

II. Technical Pain Points

A. Developing a Multimodal Sentiment Analysis System for Elderly Care

With the increasing trend of population aging, there is a growing demand among the elderly for smart home solutions. However, the current market primarily focuses on smart home products targeting younger individuals, lacking attention and consideration for the elderly. Additionally, there is a lack of emphasis on addressing their psychological well-being. Therefore, we aim to develop a comprehensive smart multimodal sentiment analysis (SMSA) system for the entire home, to meet the needs of older adults and enhance their quality of life.

This system will utilize multiple sensors to collect behavioral and emotional data of older adults within their homes.

Zexin Fu, Weilin Ruan and Songning Lai are all employees of Guangzhou Zhijia Co., Ltd., located in Guangzhou 510000, China.
Corresponding author: Zexin Fu(zexin.foo@gmail.com), Weilin Ruan(weilinruan@gmail.com) and Songning Lai(sonly@mail.sdu.edu.cn).
The three authors contributed equally and in no particular order.

Through the analysis and processing of this data, the system can automatically recognize the emotional states of the elderly, such as happiness, anger, sadness, anxiety, and loneliness, and respond accordingly. For instance, when the system detects feelings of loneliness, it can automatically play comforting music or remind the elderly to engage in video calls with their family members.

B. Enhancing Interpretability in Multimodal Sentiment Analysis for Smart Home Systems

The motivation behind integrating interpretable research in multimodal sentiment analysis within our smart home system is twofold. Firstly, while multimodal sentiment analysis has shown significant progress in accuracy, many models are considered opaque due to their complex and deep structures, limiting their interpretability. Interpretable models are important for establishing trust by enabling users to understand and calibrate their reliance on the model's predictions. Improved interpretability in multimodal sentiment analysis is a novel research area that addresses this challenge.

Secondly, the interpretability of multimodal sentiment analysis models holds practical benefits. Users often need to comprehend how the model reaches its conclusions to better understand and utilize it. For example, in social media analysis, understanding why a post is identified as positive or negative aids in understanding the sentiment towards a specific topic. In our smart home system, an interpretable sentiment analysis model enhances the provision of refined services. Moreover, interpretability contributes to model reliability, robustness, fairness, and transparency by enabling error analysis, model improvement, and bias detection.

C. The computational bottleneck and solutions for whole-house smart systems

The level of intelligence in whole-house smart systems is limited by the computational power of various computing platforms. This product targets the elderly population and aims to use televisions as the central platform for computing and interaction, considering their familiarity with TVs. However, current televisions often employ low-performance CPUs that cannot efficiently execute AI algorithms. Edge sensing products are limited by their size, and their computational power needs improvement as well. These two factors significantly impact the intelligence experience in the whole house.

This system focuses on addressing these two challenges by categorizing the sensing systems into three classes based on their computational requirements. It then designs a heterogeneous System-on-Chip (SoC) architecture and AI-specific hardware accelerators to provide ample computational power support for AI applications in the whole-house smart system.

III. Sensing and Computing System for AI Applications

A. Sensing and Computing in Whole-House Smart Systems

For the care of the physical and mental well-being of the elderly, our system incorporates a wide range of sensors. The computing devices execute algorithms to utilize the sensor information for health analysis. On one hand, the sensor information is used for physical health monitoring. For example, temperature and heart rate information can be used to detect abnormalities, video information can be used for fall detection, and audio information can be used for distress call detection. On the other hand, this information, along with the derived analysis of the physical health status, can serve as multimodal inputs for analyzing the psychological well-being of elderly individuals.

Based on the design requirements for different application scenarios (such as size limitations), algorithm complexity, and computational needs, we categorize the sensing and computing products in the whole-house smart system into three categories: ultra-compact sensing systems with high portability, edge computing systems with certain intelligence, and high-performance terminal presentation systems. These three categories of systems will employ different product designs, hardware architectures, deploy various types of intelligent algorithm applications, and communicate through wireless WiFi or wired Ethernet. We have already introduced three products: SmartClip, SmartAgent, and SmartVision, which cover these three categories. Next, we will provide an overview of the corresponding technical solutions for each category.

B. SmartClip: Ultra-Compact Sensing System

1) Technical Solution Design: This system is based on the STM32F103 as the control chip, using the Wildfire STM32 Nucleo development board. The ADI MAX30205 temperature sensor and MAX30102 heart rate sensor are inserted into the development board, forming a complete system.

2) Intelligent Applications: Abnormal Body Temperature and Heart Rate Detection

The STM32 chip reads data from the temperature and heart rate sensors, which is then transmitted to the TV terminal via a WiFi module. The TV terminal displays the elderly person's temperature and heart rate data and determines whether there are any abnormalities based on predefined thresholds.

C. SmartAgent: Edge Intelligent System

1) Technical Solution Design: This system is based on the Diligent ZYBO Z7-20 FPGA development board. The core FPGA chip is the Xilinx ZYNQ 7020, and its FPGA resources are shown in Table I. The system is equipped with an OV5640 monocular camera, a microphone, and external speakers, forming a complete system.

TABLE I
FPGA Resources for ZYNQ 7020 and 7100

	Logic Cells	Block RAM	DSP Slices	I/O Pins
ZYNQ 7020	85K	4.9Mb	220	200
ZYNQ 7100	444K	26.5Mb	2020	400

We have designed a dedicated lightweight AI acceleration chip for the SmartAgent edge computing intelligent system, enabling efficient local AI algorithm inference. This allows for a range of intelligent applications, including fall detection,

voice assistant wake-up, distress call detection, and more. The SmartAgent edge computing system transfers data to the SmartVision TV terminal via a wired Ethernet connection, providing a low-latency and stable connectivity experience.

2) Intelligent Application 1: Fall Detection

In this system, fall detection is based on a pose estimation algorithm that recognizes the skeletal keypoints of the human body. We have chosen CMU's OpenPose [48] as the core algorithm for fall detection. OpenPose is the world's first real-time multi-person 2D pose estimation algorithm based on deep learning. It utilizes VGG19 as the backbone network, and its network architecture is illustrated in Figure 2.

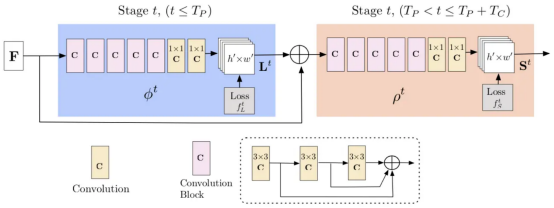

Fig. 2. The OpenPose Network Architecture [49]

Through the use of human keypoint detection techniques, OpenPose can accurately locate and track human bodies in real time, as shown in Figure 3. It utilizes discriminative methods such as the minimum potential energy to connect the keypoints and form a skeletal model. By leveraging specific keypoints identified by OpenPose, we can calculate corresponding angles, heights, and other information. Based on predefined decision criteria, we can determine whether a fall has occurred.

Fig. 3. OpenPose Demo

A demonstration of the fall detection application is shown in Figure 4.

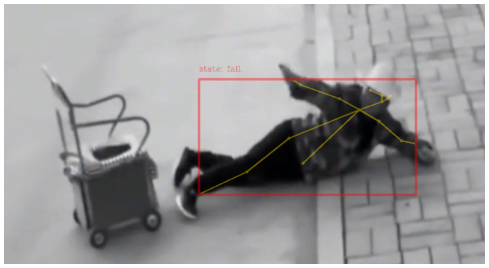

Fig. 4. Fall Detection Demo

3) Intelligent Application 2: Keyword Wake-up and Distress Call Detection

Both the keyword wake-up and distress call detection functions in this system are based on Keyword Spotting, using the MicroNets algorithm [50] as the core algorithm. MicroNets is a resource-constrained design algorithm that utilizes Differentiable Neural Architecture Search (DNAS) to discover models with low memory usage and low operations. The network architecture of MicroNets is shown in Figure 5, consisting primarily of Convolutional (Conv) layers, Average Pooling (AvgPool), and MobileNet V2's inverted bottleneck [64].

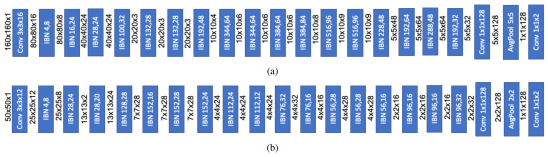

Fig. 5. Network Architecture of MicroNets [50]

During the algorithm execution, the audio is first segmented into 1-second-long segments. Each segment is then transformed into a two-dimensional feature using Mel-frequency cepstral coefficients (MFCC). The Convolutional Neural Network (CNN) is then employed to classify the two-dimensional features and provide the probabilities for several predefined keywords. For the keyword wake-up application, the keyword can be set as "Hello, AI," which is a binary classification problem. For the distress call detection application, common distress phrases such as "Help," "Is anyone there?," and "Please help me" can be used as keywords.

For the implementation of the voice assistant, SmartAgent acts as the proxy, accepting voice input and providing voice feedback through its audio module. When the wake-up keyword is detected, SmartAgent transmits the voice stream via Ethernet to the SmartVision terminal for speech recognition, answering, and voice synthesis. The resulting voice stream is then transmitted back to SmartAgent, which outputs the voice response.

As an example, the demonstration of the keyword wake-up application is shown in Figure 6.

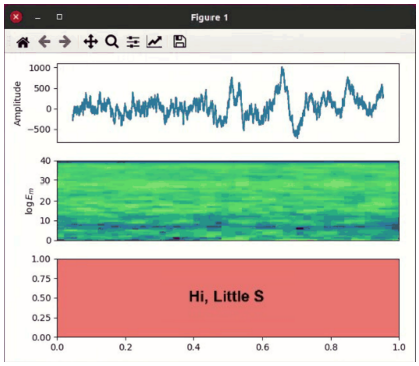

Fig. 6. Keyword Wake-up Demo

4) Hardware Acceleration Support for AI Algorithms: Given that deep learning algorithms share common operations (such as convolution, fully connected layers, pooling, and Transformers), we have designed and implemented a dedicated accelerator called CNNLighter on the FPGA based on the computational and memory access characteristics of these operations. This accelerator is integrated with the CPU on-chip as part of the system.

Considering the size and power constraints of the SmartAgent edge computing system, we have developed a lightweight version of CNNLighter called CNNLighter_Lite. Additionally, a low-power CPU, the Hummingbird E203 CPU [51], is selected. The specific details of the hardware acceleration design are described in Section V.

D. SmartVision: High-Performance Terminal System

1) Technical Solution Design: The SmartVision high-performance terminal system is based on the RJIBI FACE Z7 FPGA development board, developed in collaboration with TV manufacturers. The core chip of the development board is the Xilinx ZYNQ 7100, which offers FPGA resources as shown in Table I.

We have designed a dedicated AI acceleration chip for the SmartVision high-performance terminal system, enabling fast local AI algorithm inference. This supports a range of intelligent applications, including voice assistants and multimodal sentiment analysis. Depending on the requirements of different applications, the SmartVision high-performance terminal system is connected to the sensor system via wireless WiFi or wired Ethernet for data transmission.

2) Intelligent Application 1: Voice Assistant

The implementation of the voice assistant involves five main steps: keyword wake-up (Keyword Spotting), automatic speech recognition (ASR), natural language understanding (NLU), natural language generation (NLG), and text-to-speech (TTS). We deploy the keyword wake-up module on the SmartAgent edge computing system for user interaction, while the SmartVision high-performance terminal system serves as the central processing unit to handle the remaining four tasks. We utilize the open-source wukong-robot project to build the voice assistant, providing hardware acceleration support at the underlying level. wukong-robot is characterized by its modular and customizable features, and its functional framework is shown in Figure 7.

The demonstration of the voice assistant application is shown in Figure 8.

3) Intelligent Application 2: Multimodal Sentiment Analysis

For a detailed introduction to the multimodal sentiment analysis system, please refer to Section III. The explainability of the system is discussed in Section IV.

4) Hardware Acceleration Support for AI Algorithms: We have designed and implemented a dedicated AI accelerator, CNNLighter, on the FPGA. This accelerator is integrated with the CPU on-chip as part of the system.

We deploy a full-featured CNNLighter on the SmartVision high-performance terminal system, utilizing the LowRisc Ariane CPU [52] and adding instruction fusion features to its microarchitecture to enhance performance. The specific details of the hardware acceleration design are described in Section V.

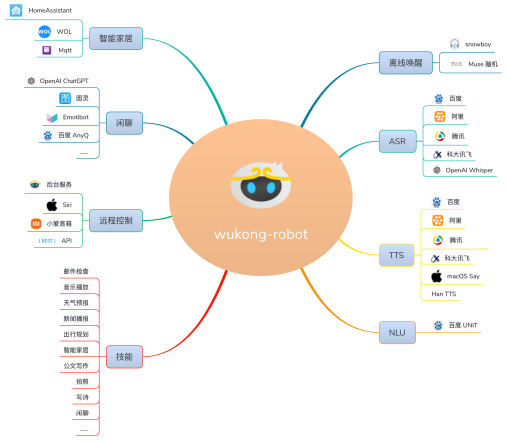

Fig. 7. Functional Framework of wukong-robot

Fig. 8. Voice Assistant Demo

IV. IMSAS

A. Related Works

Multimodal Sentiment Analysis (MSA) MSA aims to predict sentiment polarity and sentiment intensity under a multimodal setting (Morency et al.,2011). MSA research could be divided into four groups. The first is multimodal fusion. Early works of multimodal fusion mainly operate geometric manipulation in the feature spaces (Zadeh et al.,2017). The recent works develop the reconstruction loss (Hazarika et al., 2020), or hierarchical mutual information maximization (Han et al., 2021) to optimize multimodal representation. The second group focuses on modal consistency and difference through multi-task joint learning (Yu et al., 2021a) or translating from one modality to another (Mai et al., 2020). The third is multimodal alignment. Tsai et al. (2019a) and Luo et al. (2021) leverage cross-modality and multi-scale modality representation to implement modal alignment, respectively. Lastly, studies of multimodal context integrate the unimodal context, in which Chauhan et al. (2019) adapts context-aware attention, Ghosal et al. (2018) uses multi-modal attention, and Poria et al. (2017) proposes a recurrent model with multi-level multiple attentions to capture contextual information among utterances.

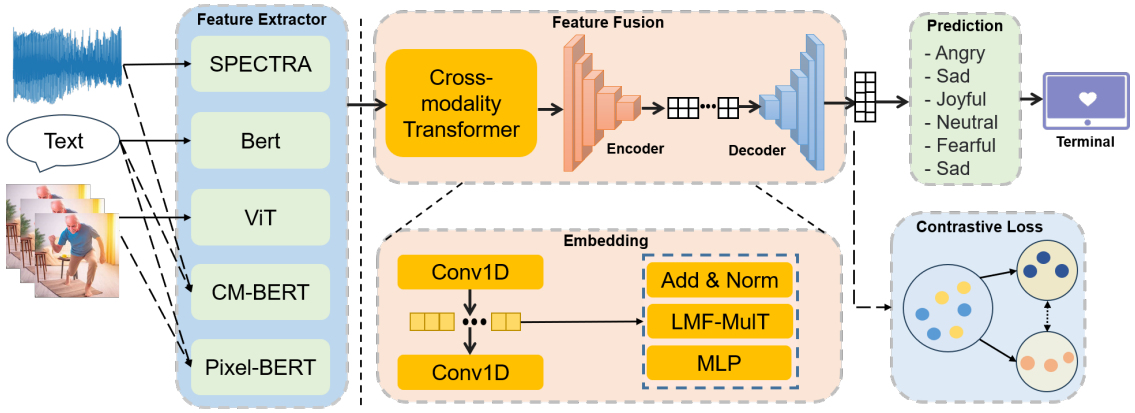

Fig. 9. The overall architecture of interpretable multimodal sentiment analysis models

Emotion Recognition in Conversations (ERC) With growing research interest in dialogue systems (Dai et al., 2021, 2020a,b; Lin and Xu, 2019a,b; Lin et al., 2020; Zhang et al., 2022b), how to recognize the predefined emotion in the conversation has become a research hotspot. Meanwhile, with the rise of multimodal machine learning (Mao et al., 2022; Yuan et al., 2021; Yu et al., 2021b; Zhang et al., 2022a; Lin et al., 2022), the studies of ERC have been extended to multimodal paradigm. The multimodal emotion recognition in conversation gained great progress. The research direction could be categorized into multimodal fusion, context-aware models, and incorporating external knowledge. Hu et al. (2022, 2021c); Joshi et al. (2022) adopt graph neural networks to model the inter/intra dependencies of utterances or speakers. For context incorporation, Sun et al. (2021); Li et al. (2021a); Ghosal et al. (2019) model the contexts by constructing graph structure, and Mao et al. (2021) introduces the concept of emotion dynamics to capture context. Moreover, some advancing ERC works incorporate external knowledge, such as transfer learning (Hazarika et al., 2019; Lee and Lee, 2021), commonsense knowledge (Ghosal et al., 2020), multi-task learning (Akhtar et al., 2019), and external information (Zhu et al., 2021) to solve ERC task.

B. Proposed Approach

1) Overall Architecture: We designed an interpretable multimodal sentiment analysis system as shown in the figure above. It is divided into three main blocks: feature extractor, feature fusion, and prediction model. First, we want to perform feature extraction process separately for each modality and for the information aligned between modalities. [14] The various feature extractors are selected by using the model with the highest performance in its modal domain as far as possible to achieve the global optimal feature extraction scheme. [15] The feature fusion is mainly processed by the cross-modal Transformer, which aims to make the modal fusion results retain the maximum local features with high accuracy. The prediction results are then obtained by an encoder-decoder. In addition, the model parameters are optimized by minimizing the loss function during model training so that it can accurately predict and analyze sentiment. [16] The contrast loss function is used here. The system covers the whole process from data pre-processing to model training and application, and the prediction results are finally delivered to the TV terminal. [17]

2) Data Alignment: Multimodal data usually contains information from different modalities, such as text, image, audio, etc. In our system, it mainly comes from video and audio captured by cameras and heart rate and temperature provided by wearable devices, which have clear temporal and spatial correlation and consistency. In order to better utilize the relevant information between different modal data in subsequent fusion and analysis, we can choose to use data alignment. The process of data alignment aims to synchronize the data from different modalities and ensure their consistency in time and space. Through data alignment, correspondences between modalities can be established to better understand and utilize multimodal data. For example, the SPECTRA model we currently choose to use can be used to capture the alignment of speech and text, and it works very well to pre-train this model through speech-text dialogue scenarios.

3) Feature-Extraction: We wish to divide the feature extraction task for each modality into two parts, intra-module and inter-module, and select the feature extraction modules separately based on the performance of various feature representations of the current SOTA models. [21] For single-modal text, speech, and images, we provide three sub-modules for initial extraction: (1) the latest speech-text dialogue pre-training model SPECTRA to train dialogue scenarios and use time-unknown predictions to capture speech-text alignment, which is well suited for sentiment analysis of verbal interactions in smart homes; [13](2) the multimodal BERT architecture has been performing well in text sentiment analysis tasks and has a wide range of applications in the pre-training process; (3) Vision Transfomer extracts visual features at the block level and enhances the model's sentiment analysis in video modality, and by fusing the video features extracted by ViT with features from other modalities, an integrated multimodal representation can be constructed to better understand the relationship between video and other modalities, thus improving the performance of sentiment analysis. [19]

Multimodal fusion has been proven to greatly improve training efficiency and model robustness by exploiting the redundancy and complementarity of multiple modalities. There-

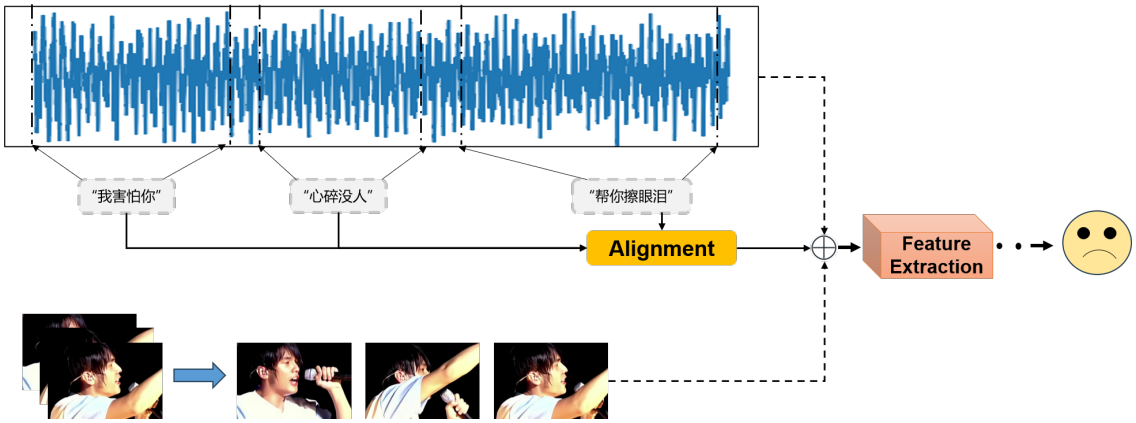

Fig. 10. Schematic diagram of how to align text information with audio or video information, here the material is taken from Jay Chou's "Dark Sign"

fore, we mainly adopt two models with good performance to extract combined speech-text modality and combined image-text modality: (1) CM-BERT is a cross-modal BERT model that relies on the interaction of text and audio modalities to fine-tune the pre-trained BERT model and thus can be used as feature extraction for text-audio modality combinations. [18](2) Pixel-BERT aligns image pixels with text via deep multimodal deformers to jointly learn visual and linguistic embeddings in a unified end-to-end framework. SOTA results were also achieved in training for feature extraction of image-text combinations. [38]

4) Cross-modality Transformer: The feature representations obtained from the data of various modalities processed by the feature extractor can be fused by Cross-modality Transformer to obtain an embedding that greatly preserves the information of each modality. [22] Subsequently, after mapping by encoder-decoder models (LSTM, Transformer and MLP are common) [23], the features are projected to the corresponding sentiment classification. Adequate information interaction and parameter updates are ensured between the working modules, and very good prediction results can be achieved.

The internal part of the Cross-modality Transformer is composed of several parts. First, a convolution operation is performed on the features to achieve cross-membrane state integration and information interaction and to reduce the network parameters. The one-dimensional convolution operation results in feature data that is more representative of local information, and after some common fusion methods, such as LMF-MulT [31], a lower-dimensional embedding vector can be obtained. The low-dimensional feature representation is subjected to regularization operations including batch normalization and dropout, which can improve the robustness and generalization ability of the model and be used to reduce overfitting and optimize the model performance. Finally, the multilayer perceptron (MLP) module is used to further map and model the embedding vectors, which can better capture the high-level semantics and relevance in multimodal data. [**?**] Through the combination and processing of the above components, Cross-modality Transformer is able to achieve cross-modality feature integration, interaction and learning, and overall improve the performance of multimodal sentiment analysis for specific downstream tasks.

5) Contrastive Loss: Contrastive Loss is a loss function for multimodal representation learning. Its goal is to train a model by maximizing the similarity between samples of the same class and minimizing the similarity between samples of different classes. Using Contrastive Loss in a multimodal task requires the fusion of features and then constructing sample pairs to compute similarity. Then, the model is trained by minimizing the Contrastive Loss so that we can learn a more efficient multimodal representation.

The process of constructing positive and negative sample pairs is as follows. A pair of samples are selected from the training data, one of which is the anchor sample (anchor) and the other is either a positive sample (positive) or a negative sample (negative). The anchor sample and the positive sample are from the same category, while the anchor sample and the negative sample are from different categories. We need to calculate the dissimilarity between the following two pairs of samples to measure the distance or similarity between the samples, taking the cosine similarity as an example, the formula is as follows:

$$sim_p = cos(\theta_1) = \frac{\hat{anchor} \cdot \hat{positive}}{(|\hat{anchor}| * |\hat{positive}|)}$$
$$sim_n = cos(\theta_2) = \frac{\hat{anchor} \cdot \hat{negative}}{(|\hat{anchor}| * |\hat{negative}|)}$$

The boundaries of similarity and dissimilarity are determined by defining thresholds or hyperparameters, and we usually calculate Contrastive Loss using cosine similarity or Euclidean distance. during the training process, optimization algorithms such as gradient descent are used to continuously adjust the model parameters while The effect of minimizing Contrastive Loss is achieved by making the similarity of samples from the same category as close to 1 as possible, and the similarity of samples from different categories as close to 0 as possible.

This method aims to allow the similarity between different modalities to be enhanced and the similarity between samples of the same category to be boosted, thus improving the performance and generalization ability of the multimodal task.

C. Experiments

1) Datasets: The datasets we used include the multimodal opinion-level sentiment intensity dataset MOSI and the multimodal opinion sentiment and sentiment intensity MOSEI, where MOSI contains 2,199 discourse video clips, each manually annotated with a sentiment score ranging from - to +3 to indicate the sentiment polarity of the segment and the relative sentiment intensity of the relative sentiment intensity.MOSEI is an upgraded version of MOSI with sentiment and sentiment annotations. upgraded version of MOSI with sentiment and mood annotations. mOSEI contains 22,856 movie review clips from YouTube. most existing studies use only the sentiment annotations of MOSEI, which are multi-labeled, so we did not use its sentiment annotations even though they are available.

Note that there is no overlap between MOSI and MOSEI, and the data collection and annotation processes for these two datasets are independent. Following previous work, we selected six emotions for sentiment identification, including joy, sadness, anger, neutrality, excitement, and dismay. mELD contains 13,707 video clips of multiparty conversations whose labels follow the six universal emotions, including joy, sadness, fear, anger, surprise, and disgust.

TABLE II
THE DETAILS OF MOSI AND MOSEI INCLUDING DATA SPLITTING AND THE LABELS IT CONTAINS.

	Train	Valid	Test	All
MOSI	1284	229	686	2199
MOSEI	16326	1871	4659	22856

2) Evaluation metrics: For MOSI and MOSEI, we followed previous work and used mean absolute error (MAE), Pearson correlation (Corr), seven-category classification accuracy (ACC-7), two-category classification accuracy (ACC-2), and F1 score. Classification accuracy (ACC-7), binary classification accuracy (ACC-2) and F1 scores, respectively, were calculated as F1 scores of positive/negative and non-negative/negative classifications were used as evaluation indexes.

3) Results: We compared our model with the baseline on the data sets MOSI and MOSEI, and the results are shown in the table.IMSAM significantly outperformed SOTA on MOSI, MOSEI for most metrics.IMSAM improved ACC-2 for MOSI and ACC-2 for MOSEI by 1.65% and 1.16%, and improved F1 for MOSI, F1 for MOSEI's F1 by 1.73% and 1.29%, respectively. It can be seen that the current performance of IMSAM on these two datasets has outperformed earlier models such as LMF, TFN, and MFM, illustrating the effectiveness and superiority of IMSAM in MSA tasks and demonstrating the feasibility of the entire framework of interpretable multimodal sentiment analysis for generalized sentiment analysis tasks.

4) Ablation Study: We conducted a series of ablation studies on MOSI, and the results are shown in Table. First, we eliminate one or several modalities from multimodal signals to verify the modal effects on model performance. We can find that removing visual and acoustic modalities or one of them all leads to performance degradation, which indicates that the non-verbal signals (i.e., visual and acoustic) are necessary for solving MSA, and demonstrates the complementarity among text, acoustic, and visual. We also find that the acoustic modality is more important than the visual to IMSAM. Then we eliminate module Cross-modality Transformer from IMSAM, which leads to an increase in MAE and a decrease in Corr. These results illustrate the effectiveness of Cross-modality Transformer in multimodal representation learning. Additionally, we conduct experiments to verify the impact of the dataset on IMSAM. We remove MOSEI from the training set and evaluate model performance on the MOSI test set. This removing hurts the performance, especially in metrics MAE and Corr. This result may be because the removal has reduced the information they provide for MSA task. We also remove MOSEI, resulting in poor performance in the all metrics. The proposed IMSAM is orthogonal to the existing works, and it is believed that introducing our unified framework to other tasks can also bring improvements. The experimental results verify that the data of various modalities as well as the operation of cross-modality play a key role in the training effect.

D. Conclusion

Experimental comparisons reveal that our model achieves SOTA performance for some metrics in current large public datasets such as MOSI and MOSEI, proving the efficiency and feasibility of our approach. However, because there are no large public datasets of home scenarios for pre-training, leading to the mainstream multimodal sentiment analysis models are facing the problems of insufficient stability, low accuracy and limited generalization ability. In order to better transfer the models to the task of sentiment prediction for smart home products, we find a group of elderly people living alone as a test user group, and capture and label the daily home life data of the elderly through sensors, cameras and wearable devices, etc., and use them as scenario-specific requirement datasets in the current model training. We are still actively working on data collection and model enhancement, and believe that these efforts will lead to a steady improvement in model performance and better future prospects for the smart home market.

V. INTERPRETABILITY OF SENTIMENT ANALYSIS SYSTEM

Multimodal sentiment analysis has made significant progress in the community, and many published models have achieved sufficient accuracy on public datasets. However, due to their complex and deep structures, they are often considered as opaque models that are difficult to understand and interpret. There are many reasons why interpretable models are important, such as establishing trust in the model by creating calibration, which means understanding when we should trust the model. Making multimodal sentiment analysis models more interpretable is currently a very novel research topic.

The importance of studying the interpretability of multimodal sentiment analysis systems is as follows:

The interpretability of multimodal sentiment analysis models is important for the application of the models. In practical applications, users often need to understand how the model arrives at its conclusions in order to better understand and use

	MOSI					MOSEI				
Method	**MAE↓**	**Corr↑**	**ACC-7↑**	**ACC-2↑**	**F1↑**	**MAE↓**	**Corr↑**	**ACC-7↑**	**ACC-2↑**	**F1↑**
LMF	0.917	0.695	33.2	-/82.5	-/82.4	0.623	0.7	48.0	-/82.0	-/82.1
TFN	0.901	0.698	34.9	-/80.8	-/80.7	0.593	0.677	50.2	-/82.5	-/82.1
MFM	0.877	0.706	35.4	-/81.7	-/81.6	0.568	0.703	51.3	-/84.4	-/84.3
ICCN	0.862	0.714	39.0	-/83.0	-/83.0	0.565	0.704	51.6	-/84.2	-/84.2
MulT	0.861	0.711	-	81.5/84.1	80.6/83.9	0.580	0.713	-	-/82.5	-/82.3
MISA	0.804	0.764	-	80.79/82.10	80.77/82.03	0.568	0.717	-	82.59/84.23	82.67/83.97
MAG-BERT	0.712	0.796	-	84.20/86.10	84.10/86.00	-	-	-	84.70/-	84.50/-
MMIM	**0.7**	0.8	46.65	84.14/86.06	84.00/85.98	0.526	0.772	**54.24**	82.24/85.97	82.66/85.94
IMSAM	0.691	**0.809**	**48.68**	**85.85/86.9**	**85.83/86.42**	**0.523**	**0.773**	54.39	**85.86/87.50**	**85.79/87.46**

TABLE III
IMSAM COMPARES THE PREDICTION RESULTS OF OTHER MODELS ON BOTH MOSI AND MOSEI DATASETS. WHERE UNDERLINE INDICATES THE CURRENT SOTA EFFECT AND BLANK INDICATES THAT THE AUTHORS HAVE UPDATED THE EXPERIMENTAL RESULTS.

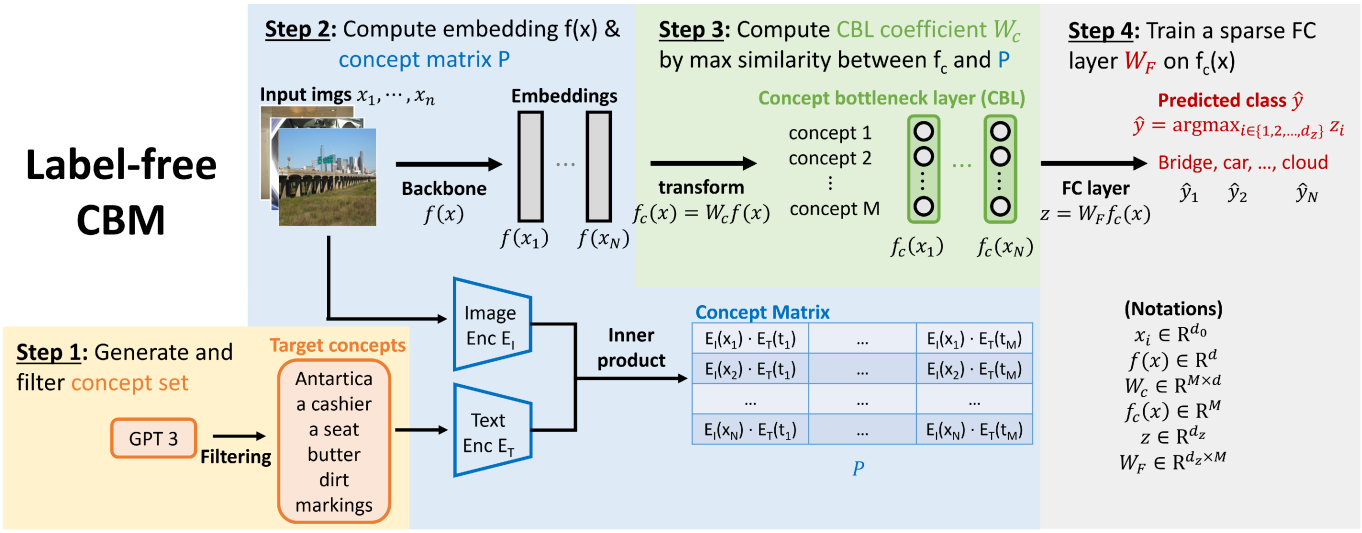

Fig. 11. Overview of the Label Free Concept Bottleneck Model

	MAE	Corr	ACC	F1
IMSAM	**0.691**	**0.809**	**85.85/86.9**	**85.83/86.42**
V	0.714	0.798	84.37/85.37	84.71/85.78
A	0.719	0.794	83.82/85.20	83.86/85.69
T	0.722	0.785	85.13/86.59	85.03/86.37
MOSEI	0.775	0.727	80.68/81.22	81.35/81.83

TABLE IV
RESULTS OF ABLATION EXPERIMENTS. WHERE V AND A DENOTE REMOVAL OF VISUAL OR AUDIO MODAL DATA AND T DENOTES REMOVAL OF THE CROSS-MODAL TRANSFORMER MODULE.

the model. For example, in social media analysis, users may need to know why a particular post is identified as positive or negative in order to better understand the sentiment towards a certain topic. In our company's product, the interpretability of the sentiment analysis system can also help provide more refined services. Therefore, the interpretability of multimodal sentiment analysis models can help users better understand and apply the models.

Secondly, the interpretability of multimodal sentiment analysis models is important for improving the reliability and robustness of the models. In practical applications, models may face various challenges such as data missing or data noise. If the models lack interpretability, we cannot determine the reasons for errors when they occur, and it is difficult to make corrections to the models. Therefore, the interpretability of multimodal sentiment analysis models can help us better understand the working principles of the models and make improvements to them.

Finally, the interpretability of multimodal sentiment analysis models is important for improving the fairness and transparency of the models. In practical applications, we often need to ensure that the models are fair and transparent. If the models lack interpretability, we cannot determine whether the models have biases or discrimination. Therefore, the interpretability of multimodal sentiment analysis models can help us better understand the decision-making process of the models and ensure that the models are fair and transparent.

In conclusion, the interpretability of multimodal sentiment analysis models is important for improving the application, reliability, robustness, fairness, and transparency of the models. Therefore, when developing multimodal sentiment analysis models, we should pay attention to their interpretability and take measures to improve it.

The Concept Bottleneck Model (CBM) is a popular approach that creates more interpretable neural networks by hiding neurons in hidden layers. In our multimodal sentiment analysis model, the primary method used is extracting features from the end layer to create an interpretable neural network. However, existing CBMs and their variants have two key limi-

tations: firstly, they require collecting labeled data for each predefined concept, which is time-consuming and labor-intensive; secondly, the accuracy of CBMs is often noticeably lower than standard neural networks, especially on more complex datasets. This poor performance poses obstacles to adopting CBMs in practical applications. Given these challenges, we adopted Label Free CBM, which is a versatile framework that can transform any neural network into an interpretable CBM without the need for labeled concept data while maintaining high accuracy. Our unlabeled CBM has many advantages - it is scalable and efficient, requiring minimal human effort to automatically train it on new datasets.

A. Related Work

More interpretable final layer: [4] proposes making the FC layer sparse, and develop an efficient algorithm for doing so. They show that sparse models are more interpretable in many ways, but it still suffers from the fact the previous layer features are not interpretable. NBDT citewan2020 propose replacing the final layer with a neural backed decision tree for another form of more interpretable decisions. Other approaches to make NNs more interpretable include Concept Whitening [6]and Concept Embedding Models [7].

CBM: Most related to our approach are Concept Bottleneck Models [**?**], [8] which create a layer before the last fully connected layer where each neuron corresponds to a human interpretable concept. CBMs have been shown to be beneficial by allowing for human test-time intervention for improved accuracy, as well as being easier to debug. To reduce the training cost of a CBM, a recent work [9] proposed Post-Hoc CBM that only needs to train the last FC layer along with an optional residual fitting layer, avoiding the need to train the backbone from scratch. This is done by leveraging Concept Activation Vectors (CAV) [10] or the multi-modal CLIP model [11]. However, the post-hoc CBM does not fully address the problems of the original CBM as using TCAV still requires collecting annotated concept data and their use of CLIP model can only be applied to if the NN backbone is the CLIP image encoder. Additionally, the performance of post-hoc CBMs without uninterpretable residual fitting layers is often significantly lower than the standard DNNs. Similarly, an earlier work Interpretable Basis Decomposition [12] proposes learning a concept bottleneck layer based on labeled concept data for explanable decisions, even though they do not call themselves a CBM.

B. Interpretability Model Explained in Detail

In this section, we elaborate on Label Free CBM, which builds a Conceptual Bottleneck Model (CBM) in an automated, scalable, and efficient manner and addresses the core limitations of existing CBMS. Given a multimodal sentiment analysis network backbone, Label Free CBM converts the backbone into an interpretable CBM without concept labels, as shown in step 1 of 11 by the following 4 steps: Create an initial concept set and filter unwanted concepts; Step 2: Calculate the embedding of the concept matrix on the backbone and training data set; Step 3: Learn to project weight Wc to create Conceptual Bottleneck Layer (CBL); Step 4: Learn sparse final layer weight WF for prediction.

1) Step 1: Concept set creation and filtering: In this step, we will describe how to create a concept set to serve as the basis of human-interpretable concepts in the Concept Bottleneck Layer. This step consists of two sub-steps: **Initial concept set creation and Concept set filtering**.

Initial concept set creation: A concept set refers to the set of concepts represented in the Concept Bottleneck Layer. In the original CBM paper [1], this is decided by domain experts as the set of concepts that are important for the given task. However, since our objective is to automate the entire process of generating CBMs, we don't want to rely on human experts. Instead, we propose generating the concept set via GPT-3 [2] using the OpenAI API. Somewhat surprisingly, GPT-3 has a good amount of domain knowledge of which concepts are important for detecting each class when prompted in the right way. Specifically, we ask GPT-3 the following: - List the important characteristics that identify emotions(such as anger). - List the most common things that accompany an emotion(such as anger).

Concept set filtering: Next we employ several filters to improve the quality and reduce the size of our concept set, as stated below:

Concept length: We delete any concept longer than 30 characters in length, to keep concepts simple and avoid unnecessary complication.

Remove concepts too similar to classes: We don't want our CBM to contain the output classes themselves as that would defeat the purpose of an explanation. To avoid this we remove all concepts that are too similar to the names of target classes. We measure this with cosine similarity in a text embedding space. In particular we use an ensemble of similarities in the CLIP ViT-B/16 text encoder as well as the all-mpnet-base-v2 sentence encoder space, so our measure can be seen as a combination of visual and textual similarity. For all datasets, we deleted concepts with similarity > 0.85 to any target class.

Remove concepts too similar to each other: We also don't want duplicate or synonymous concepts in the bottleneck layer. We use the same embedding space as above, and remove any concept that has another concept with > 0.9 cosine similarity already in concept set.

2) Step 2 and 3: Learning the Concept Bottleneck Layer (CBL): Once the concept set is obtained, the next step is to learn a projection from the backbone model's feature space into a space where axis directions correspond to interpretable concepts. Here, we present a way of learning the projection weights W_c without any labeled concept data by utilizing CLIP-Dissect [3]. To start with, we need a set of target concepts that the bottleneck layer is expected to represent as $\mathcal{C} = \{t_1, ..., t_M\}$, as well as a training dataset (e.g. images) $\mathcal{D} = \{x_1, ..., x_N\}$ of the original task. Next we calculate and save the CLIP concept activation matrix P where $P_{i,j} = E_I(x_i) \cdot E_T(t_j)$ and E_I and E_T are the CLIP image and text encoders respectively. W_c is initialized as a random $M \times d_0$ matrix where d_0 is the dimensionality of backbone features $f(x)$. The initial set $\mathcal{C}$ is created in Step 1 and the training set $\mathcal{D}$ is provided by the downstream task. We define

TABLE V
ACCURACY COMPARISON, BEST PERFORMING SPARSE MODEL BOLDED. THE RESULTS FOR OUR METHOD ARE MEAN AND STANDARD DEVIATION OVER THREE TRAINING RUNS. *INDICATES REPORTED ACCURACY.

		Dataset				
Model	Sparse final layer	CIFAR10	CIFAR100	CUB200	Places365	ImageNet
Standard	No	88.80%*	70.10%*	76.70%	48.56%	76.13%
Standard (sparse)	Yes	82.96%	58.34%	**75.96%**	38.46%	**74.35%**
P-CBM	Yes	70.50%*	43.20%*	59.60%*	N/A	N/A
P-CBM (CLIP)	Yes	84.50%*	56.00%*	N/A	N/A	N/A
(Ours)	Yes	**86.40%** ± 0.06%	**65.13%** ± 0.12%	74.31% ± 0.29%	**43.68%** ± 0.10%	71.95% ± 0.05%

$f_c(x) = W_c f(x)$, where $f_c(x_i) \in \mathbb{R}^M$. We use k to denote a neuron of interest in the projection layer, and its activation pattern is denoted as q_k where $q_k = [f_{c,k}(x_1), \ldots, f_{c,k}(x_N)]^\top$, with $q_k \in \mathbb{R}^N$ and $f_{c,k}(x) = [f_c(x)]_k$.

To make the neurons in the CBL interpretable, we need to enforce the projected neurons to activate in correlation with the target concept, which we do by optimizing W_c to maximize the CLIP-Dissect similarity between the neuron's activation pattern and the target concept. Our optimization goal is to minimize the objective L over W_c as defined in Equation (1):

$$L(W_c) = \sum_{i=1}^{M} -\text{sim}(t_i, q_i) := \sum_{i=1}^{M} -\frac{\bar{q_i}^3 \cdot \bar{P_{:,i}}^3}{||\bar{q_i}^3||_2 ||\bar{P_{:,i}}^3||_2}. \quad (1)$$

Here $\bar{q}$ indicates vector q normalized to have mean 0 and standard deviation 1, and the *cos cubed* similarity $\text{sim}(t_i, q_i)$ is simply the cosine similarity between two activation vectors after both have been normalized and raised to third power element-wise. The third power is necessary to make the similarity more sensitive to highly activating inputs. As this is still a cosine similarity, it takes values between $[-1, 1]$. We optimize $L(W_c)$ using the Adam optimizer on training data $\mathcal{D}$, with early stopping when similarity on validation data starts to decrease. Finally to make sure our concepts are truthful, we drop all concepts j with $\text{sim}(t_j, q_j) < 0.45$ on validation data after training W_c. This is the 5th concept set filter from Sec 3.1. This cutoff was selected manually as a good indicator of a neuron being interpretable. During this filtering, the number of concept is reduced: $M \leftarrow M - \Delta$, where Δ is non-negative integer representing the number of concepts being removed in this step. Note that matrix W_c has also to be updated accordingly by removing the rows that corresponds to the removed concepts, and with our notation, $W_c \in \mathbb{R}^{M \times d_0}$.

3) Step 4: Learning the sparse final layer: Now that the Concept Bottleneck Layer is learned, the next task is to learn the final predictor with the fully connected layer $W_F \in \mathbb{R}^{d_z \times M}$ where d_z is the number of output classes. The goal is to keep W_F sparse, since sparse layers have been demonstrated to be more interpretable [4]. Given that both the backbone $f(x)$ and learned concept projection W_c are fixed, this is a problem of learning a sparse linear model, which can be solved efficiently with the elastic net objective:

$$\min_{W_F, b_F} \sum_{i=1}^{N} L_{ce}(W_F f_c(x_i) + b_F, y_i) + \lambda R_\alpha(W_F) \quad (2)$$

where $R_\alpha(W_F) = (1-\alpha)\frac{1}{2}||W_F||_F + \alpha||W_F||_{1,1}$, $\|\cdot\|_F$ denotes the Frobenius norm, $\|\cdot\|_{1,1}$ denotes element wise matrix norm, b_F denotes the bias of the FC layer, L_{ce} is the standard cross-entropy loss and y_i is the ground-truth label of data x_i. We optimize Equation (2) using the GLM-SAGA solver created by [4]. For the sparse models, we used $\alpha = 0.99$ and λ was chosen such that each model has 25 to 35 nonzero weights per output class. This level was found to still result in interpretable decisions while retaining good accuracy. Depending on the number of features/concepts in the previous layer this corresponds to 0.7-15% of the weights of the model being nonzero.

C. Experimental result

Dataset. To evaluate the interpretability framework, we trained an unlabeled cbm on five datasets. These data are CIFAR-10, CIFAR-100, CUB, Places365, and ImageNet.

Results of the experiment. Accuracy. The table **??** shows the performance of unlabeled CBM on all five data sets.

Show the generated concept set about emotion: I only chose 20 concepts from the emotional analogy (in fact, this is a set of thousands of concepts).

- The voice becomes shrill or high
- Facial expression is tense or distorted
- Eyes become elongated or bulging
- Gnashing of teeth or thinning of lips
- Body posture becomes stiff or tight
- No noticeable rise and fall in voice
- No noticeable changes in the eyes
- Keep a neutral or calm facial expression
- Mouth closed or slightly open
- Maintain a neutral or natural body posture
- Voice becomes shaky or tense
- Eyes become tense or widened
- Facial expression becomes nervous or frightened
- Open mouth or trembling lips
- Body posture becomes rigid or constricted
- No desire to communicate or interact with others
- Feeling isolated from your surroundings or others
- Feeling unable to fit in or fit in with your surroundings
- Not getting enough social support and connections
- Feeling not understood or cared for by others

Conceptual visualization showing interpretable multimodal sentiment analysis systems: Figure 12,13,14,15,16,17,18.

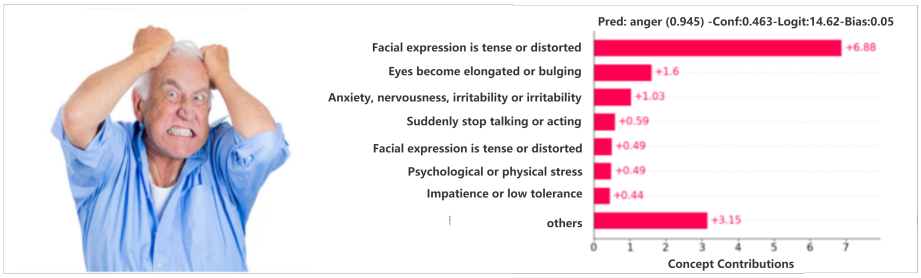

Fig. 12. A visual example of interpretable multimodal sentiment analysis 1

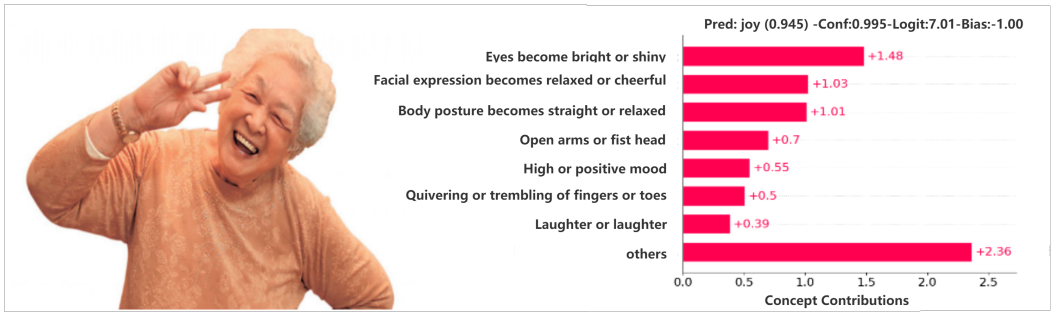

Fig. 13. A visual example of interpretable multimodal sentiment analysis 2

Fig. 14. A visual example of interpretable multimodal sentiment analysis 3

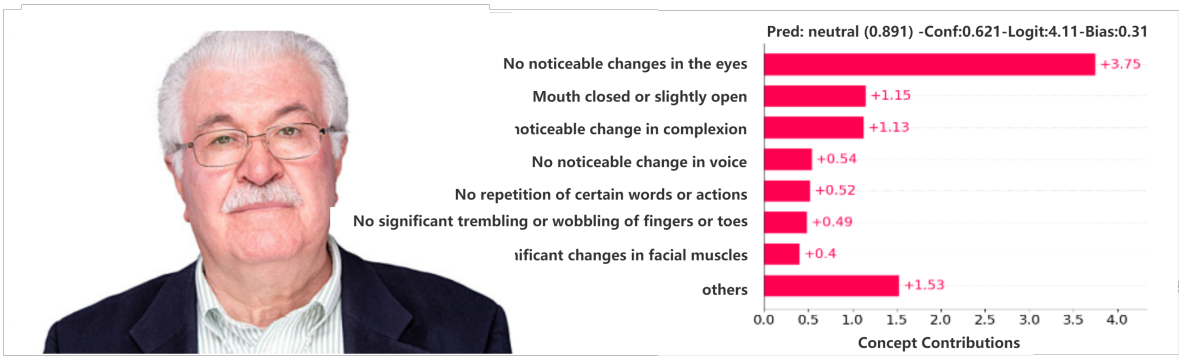

Fig. 15. A visual example of interpretable multimodal sentiment analysis 4

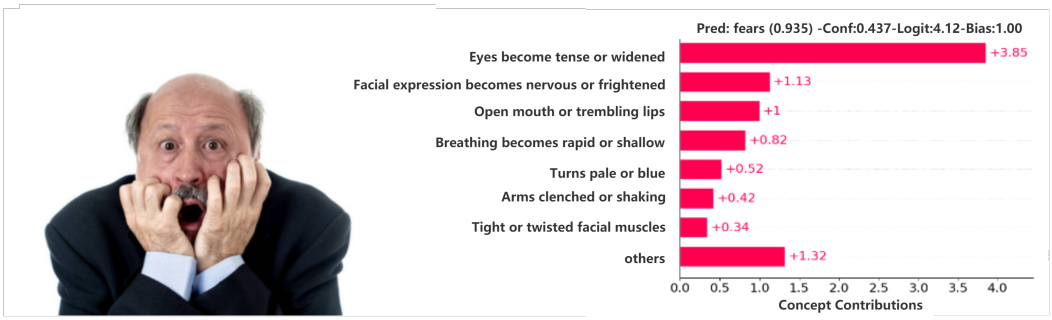

Fig. 16. A visual example of interpretable multimodal sentiment analysis 5

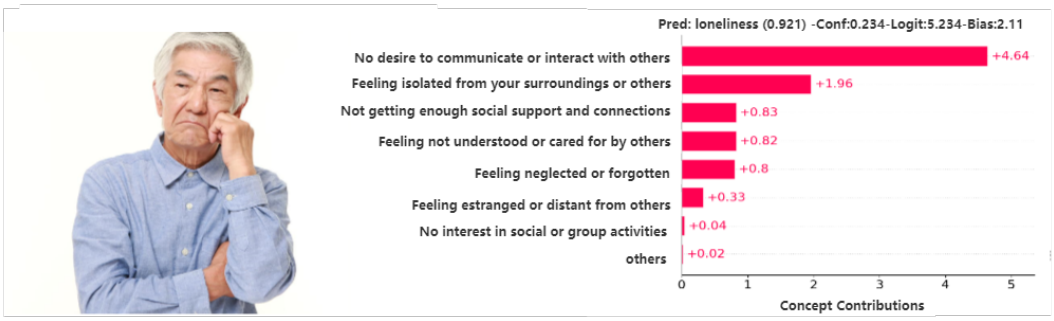

Fig. 17. A visual example of interpretable multimodal sentiment analysis 6

VI. AI Hardware Acceleration for Edge and Terminal

A. Heterogeneous SoC Architecture for Edge and Terminal

In this system, we provide hardware acceleration support for deep learning algorithms at the underlying hardware level. Due to the presence of highly parallelizable computations in the core operators of deep learning algorithms, we have designed and implemented a dedicated hardware accelerator, CNNLighter, for deep learning algorithms based on FPGA. It works in conjunction with the LowRisc Ariane CPU to form a heterogeneous on-chip system.

This heterogeneous architecture can be used for both edge devices and terminal TV devices. Considering the cost and size constraints of edge devices, we have tailored CNNLighter to create a lightweight version called CNNLighter_Lite. This accelerator consumes fewer resources and energy. Additionally, for edge devices, we have selected the low-power Bumblebee E203 processor core to further save energy.

The architecture of the SoC is illustrated in Figure 19. In this system, the upper-layer software driver and pre-processing algorithms for the hardware accelerator are executed by the CPU core. The co-accelerator accelerates deep learning algorithms, and the two components interact through the NICE interface.

We have added six custom extension instructions: FC, CONV, GEMM, POOL, BUFFER, and SYNC. These instructions are used by the CPU to control the accelerator for tasks such as fully connected operations, convolutions, general matrix multiplications, pooling computations, buffer access, and synchronization. The custom NICE instructions are encapsulated and implemented in the upper-layer software. The CPU calls the accelerator through the NICE interface to perform neural network computations. To improve data movement efficiency, we have also designed a DMA module to offload data transfer from the CPU, which is connected to a private device bus.

B. FPGA-based Deep Learning Accelerator

1) Key Features: We have designed an accelerator, CNNLighter, specifically for convolutional neural networks (CNNs). The granularity of the extended instructions, GEMM, CONV, FC, and POOL, is designed to operate on a single matrix multiplication, convolution, fully connected, or pooling operation, respectively. The accelerator supports configuration of specific parameters for each operation to ensure compatibility with any CNN architecture. It can meet the future upgrade requirements of AI algorithms. CNNLighter has the following features:

- 8x8 Processing Element (PE) array, which can be configured as a systolic array [53].
- Multi-bank memory design, supporting 8-port read/write operations.
- 16-bit fixed-point number computation.
- Mapping methods for GEMM and direct convolutions.

2) Performance Metrics:

- With a clock frequency of 100 MHz, it takes 5.86 microseconds to perform a convolution operation with an

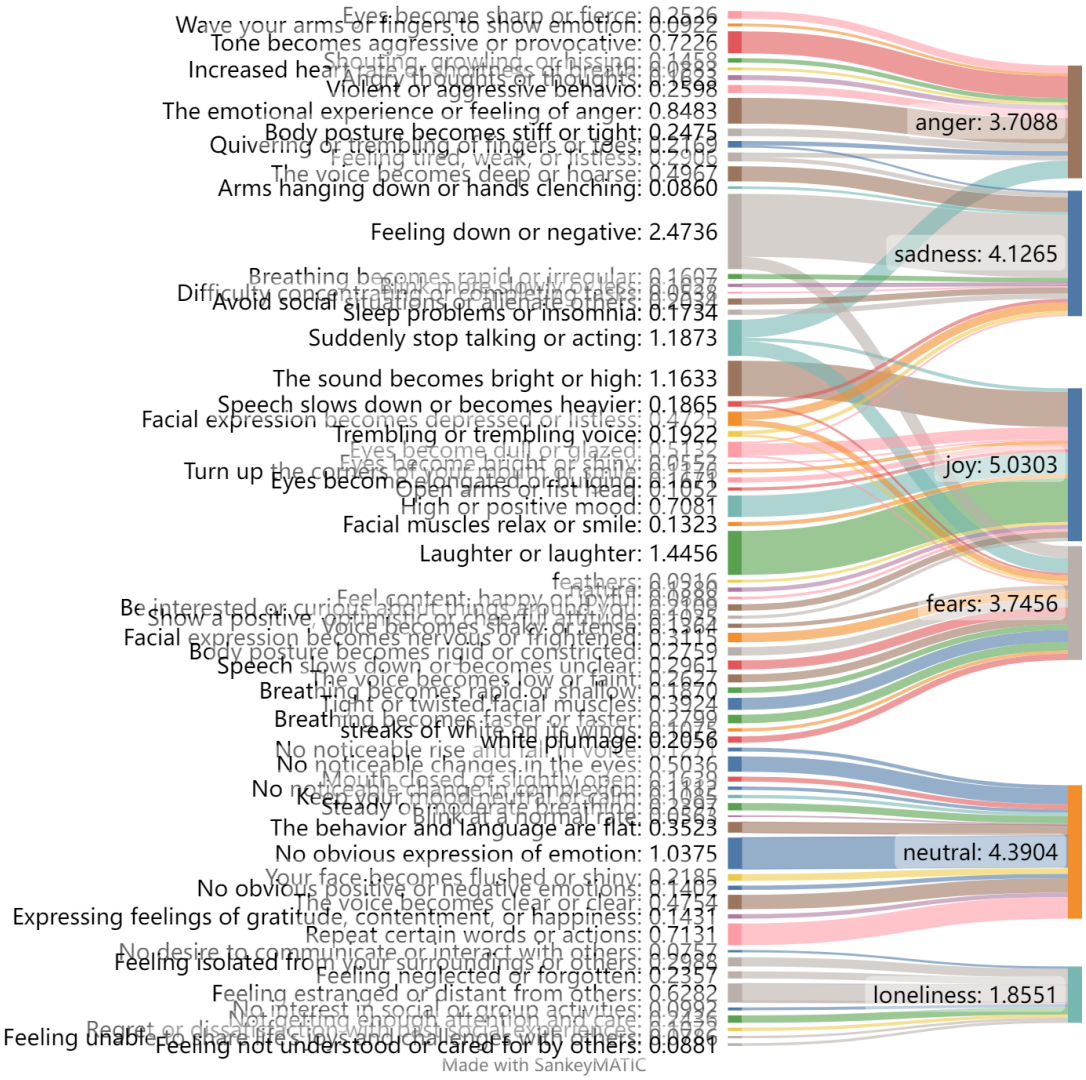

Fig. 18. A visual example of interpretable multimodal sentiment analysis 7

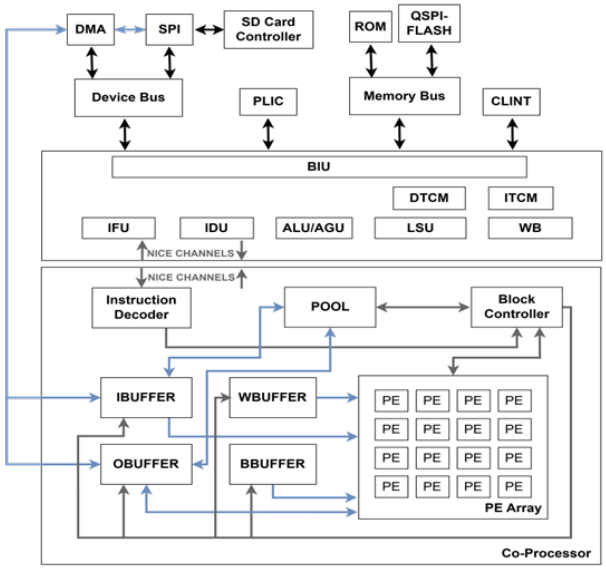

Fig. 19. SoC Architecture

input feature map size of 2x4x18, output feature map size of 16x2x16, and a convolution kernel size of 16x2x3x3.
- The core computing module consists of an 8x8 PE array with a maximum parallelism of 64.
- The memory supports 8-way parallel read/write operations.

3) Overall Architecture: The overall architecture of CNNLighter is shown in Figure 20.

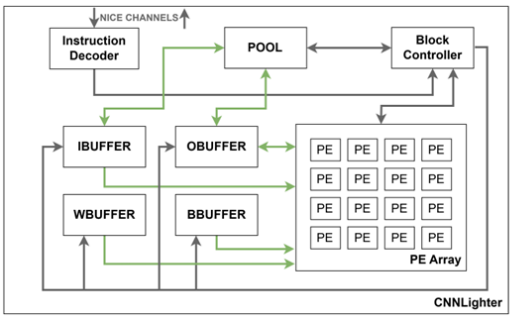

Fig. 20. Overall Architecture of CNNLighter

CNNLighter communicates with the main processor through the NICE interface. Upon receiving NICE requests, the instruction decoder decodes the instructions and passes the decoded control signals to the block controller. Based on the control signals, the block controller decomposes the instructions into multiple memory access and computation loops. Within each loop, the block controller sequentially generates parallel address signals for the four buffers and, as needed, issues memory access and address signals. Then, the block controller controls the PE array to perform computations.

4) PE Array Design: The PE array is the computing module for general matrix multiplication, convolution, and fully connected operations. It consists of 8x8 processing elements (PEs). Based on the reuse_ctrl signal from the block controller, it can be configured as a systolic array. The PE array performs computations based on various control signals from the block controller. When the block controller configures the PE array to write to the output buffer, the PE array outputs the accumulated results from the accumulation registers to the sa_wr_obuf module, which is responsible for writing the results into the output buffer. The overall structure is shown in Figure 21.

5) Multi-Bank Memory Design: The IBuffer stores the input features and supports 8-port read and 8-port write operations. It consists of 11 sets of 88x16-bit block RAMs. The OBuffer stores the output features and also supports 8-port read and 8-port write operations. It consists of 11 sets of 88x16-bit block RAMs. The WBuffer stores the weight parameters of the neural network and supports 8-port read and 1-port write operations. It consists of 8 sets of 1448x16-bit block RAMs. The BBuffer stores the bias parameters and supports 8-port read and 1-port write operations. It consists of 8 sets of 11x16-bit block RAMs.

Due to the possibility of stride greater than 1 in IBuffer and OBuffer, the number of RAMs needs to be reconsidered. It can be proved that when the stride is a factor of the number

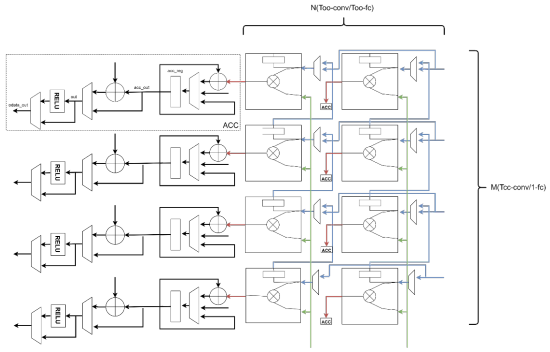

Fig. 21. PE Array Structure

of RAMs (excluding 1), there will be bank conflicts as the number of ports increases. Therefore, the number of RAMs should be chosen as a prime number larger than the stride size to avoid bank conflicts. Since the RAMs must be at least 8 (to accommodate the 8 data inputs required by the PE Array), we choose 11 as the number of RAMs for IBuffer and OBuffer. Therefore, the maximum supported convolution stride of CNNLighter is 10.

For multi-bank memory design, the critical issue is handling bank conflicts:

- The RAM count is set to 11:
  When the read/write addresses are consecutive (stride=1) and the number of ports is less than or equal to the total number of RAMs, there will be no bank conflicts when accessing the same RAM simultaneously. However, for IBuffer and OBuffer, there may be cases where the stride is greater than 1, so the number of RAMs needs to be reconsidered. It can be proved that when the stride is a factor of the RAM count (excluding 1), with an increasing number of ports, bank conflicts will occur. Therefore, the RAM count should be chosen as a prime number that is slightly larger (but not equal) than the stride size. Since the RAM count must be at least 8 (due to the requirement of feeding 8 data inputs to the PE Array simultaneously), we choose 11 as the number of RAMs for IBuffer and OBuffer. Therefore, the maximum supported convolution stride of CNNLighter is 10.
- Handling race conditions caused by RAM count being greater than the number of ports:
  Since the RAM count is greater than the number of ports, if a RAM was selected in the previous cycle but is not selected in the current cycle, and if the selection signals from the previous cycle are not cleared, it will cause a race condition in the combinatorial logic, leading to read/write errors. Therefore, it is necessary to add logic to check if a RAM was selected in the previous cycle but not in the current cycle, and if so, set the relevant selection signals to high impedance. This ensures that only one RAM is selected at a time.
- It is not allowed to simultaneously read and write in Buffer8r8w, nor to read or write data from the same BRAM at the same time. According to the memory access patterns of neural networks, it can be proved that neither

of these two situations will occur, so this limitation will not have any impact.

C. Lightweight CNNLighter_Lite

To cater to the needs of lightweight neural networks, we have tailored CNNLighter to create a more lightweight accelerator called CNNLighter_Lite. It has the following characteristics:

- The PE array does not support systolic array configuration as the inter-PE connections are removed.
- The pooling module and related operations are removed, and Stride Convolution is used as a replacement for pooling operations [62].
- It is designed for the deployment of lightweight neural networks, where all input features and parameters are stored on-chip, eliminating the need for off-chip memory access.

D. Error and Resource Utilization

Through evaluation using test cases, the average error between CNNlighter_Lite and software simulation results is 0.3546

The FPGA resource utilization for CNNlighter_Lite is shown in Figure 22.

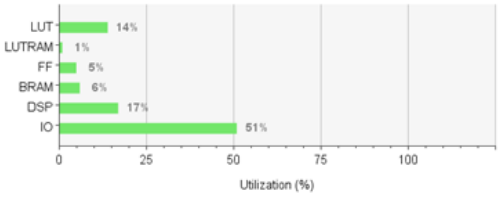

Fig. 22. Resource Utilization of CNNlighter_Lite

E. Fixed-Point Quantization Schemes

In order to save FPGA storage resources, we convert the neural network's 32-bit floating-point numbers to 16-bit fixed-point representation. We have developed a fixed-point bit-width search algorithm and tested all 15 fixed-point schemes for their accuracy compared to the 32-bit floating-point algorithm.

For each fixed-point scheme, we tested it on a set of 1000 data samples from the target algorithm. The final test results are shown in Figure 23:

Based on the test results, the scheme with 11-bit integer part and 4-bit fractional part had the lowest error. The numerical error was 6.38% and the classification error was 0.10%. It can be seen that this fixed-point bit-width has almost no impact on classification accuracy. Therefore, we have chosen the fixed-point scheme of 1-bit sign, 11-bit integer part, and 4-bit fractional part.

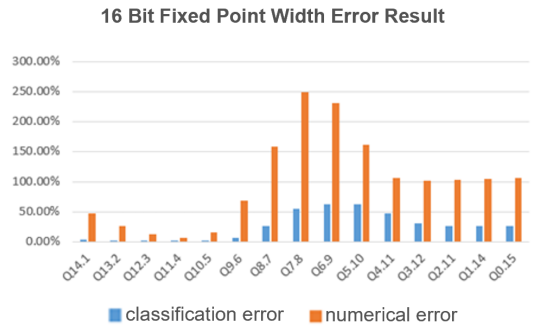

Fig. 23. Error rates for 15 fixed-point schemes

F. Instruction Fusion Technology for High-Performance CPUs

The design of high-performance processor chips involves high complexity. It requires the construction of a complete technical system to support the design process, including feature analysis of workload programs, simulator design, exploration of microarchitectural design space, performance evaluation methods, processor simulation and verification technologies, and more. To achieve powerful performance, the microarchitecture of high-performance processors generally has the following characteristics: deep pipelines to increase clock frequency, superscalar and multiple-issue to achieve higher instruction-level parallelism, out-of-order execution to overlap instructions, larger and more complex multi-level cache designs to reduce memory latency, etc. The performance of a processor can be measured using the formula shown in Figure 24.

$$\underset{\text{(secs/program)}}{\textbf{Performance}} = \frac{\text{Cycles}}{\text{Insts}} * \frac{\text{seconds}}{\text{Cycles}} * \frac{\text{Insts}}{\text{Program}}$$

Fig. 24. Processor Performance Formula [54]

From this formula, it can be seen that one approach to improving performance is to minimize the number of instructions required to execute a program, which means increasing instruction density. Instruction fusion is an important technique to achieve this goal.

Due to the differences in the execution complexity of instructions defined by the instruction set, high-performance processors often split or fuse instructions during the decoding stage before instruction dispatch. Complex instructions are split into several relatively simple instructions, while excessively simple instructions are fused into a relatively complex instruction. This ensures that the instructions entering the instruction dispatch queue have similar complexity levels, reducing bandwidth and resource wastage.

We propose a framework for instruction fusion analysis, testing, and design for the RISC-V architecture. The framework includes the analysis of fusion opportunities for different instruction pairs, the use of a simulator to generate dynamic instruction sequences of workload programs, the exploration of fusion space, and the analysis and design of fusion schemes based on the test results. Let's explore each step in detail.

1) Instruction Fusion Framework: We have developed a comprehensive design framework for instruction fusion analysis, testing, and fusion scheme design under the RISC-V architecture [56], as shown in Figure 25.

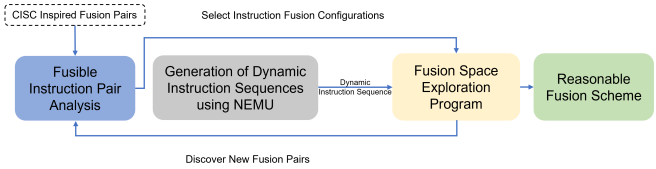

Fig. 25. Instruction Fusion Design Framework

The overall process is as follows: Firstly, this design conducts a thorough analysis of instruction pairs that offer fusion opportunities and benefits, in order to determine the available range of instruction fusion pairs. Next, the NEMU simulator [55] is used to simulate the execution of specified benchmark tests on the RISC-V processor, generating dynamic instruction sequences during the execution. This design includes a fusion space exploration program, which can simulate fusion tests for all dynamic instruction sequences under various fusion pair configurations, thereby obtaining data such as the dynamic instruction count before and after fusion. Based on this data, further analysis is conducted to design a reasonable fusion scheme tailored to the workload of this design. The subsequent sections will provide detailed explanations of each component in the framework.

Analysis of Fusion Opportunities

Instruction fusion aims to provide support for complex instruction operations in high-performance RISC-V processors through microarchitectural fusion. We analyze several instruction pairs with fusion opportunities and benefits by extracting common instructions from the CISC instruction set that do not have corresponding instructions in the RISC-V instruction set. Each instruction pair may have different composition forms, which we refer to as subclasses of instruction pairs.

In addition to manual analysis of fusion opportunities, our fusion space exploration program can discover new fusion pairs and subclasses by analyzing benchmark tests.

Generation of Dynamic Instruction Sequences using NEMU

To test the actual fusion effects, we need to obtain the dynamic execution information of workload programs, which includes all dynamic instruction sequences. Therefore, we use the NEMU simulator to simulate the execution of RISC-V processors and output the dynamic instruction sequences of the workload programs for further testing and analysis.

Different workload programs have different distributions and frequencies of instruction pairs, which will result in different fusion schemes. The NEMU simulator can run various workload programs, including public benchmarks such as Coremark and SPEC CPU 2006, as well as specific task programs.

We divide the workload programs into two categories and use different approaches to run them in NEMU. For simple embedded workloads, we compile and generate binary files to run in the bare-metal environment of NEMU. For desktop and server workloads that require an operating system environment, we provide NEMU with a proxy kernel, boot loader, and Linux kernel to boot the Linux operating system and execute the workloads. Some workload programs require long execution times, and we use NEMU's Simpoint Checkpoint feature to analyze and select representative fragments from large workload programs. Running these fragments can replace the execution of the complete programs, greatly reducing the analysis and design cycle of fusion schemes.

Testing Fusion Space Exploration Program

We have developed a fusion space exploration program that can test and analyze the fusion effects of different instruction pairs. The program allows for the enablement or disablement of each class of instruction pairs and their subclasses.

By running this program, we can analyze the percentage reduction in dynamic instruction count and the distribution of different classes of instruction pairs. For each class of instruction pair, we provide analysis for matching failures due to register and operand mismatches, which can help discover new fusion subclasses. For instruction pairs with high frequencies, we also support analysis of non-contiguous fusion opportunities to further improve fusion effects (the current version only supports non-contiguous fusion for load-pair/store-pair).

Analysis and Design of Fusion Schemes

Based on the test results and the output information from the fusion space exploration program, we can determine the fusion range for different instruction pairs and provide guidance for microarchitecture design.

2) Experiment: We tested the fusion effects of eight standard subclasses and general subclasses on the Coremark benchmark. The dynamic instruction count was used as the testing metric.At the same time, this design also tested the storage optimization effects brought by enabling the compressed instruction set extension, with the measurement metric being the number of fetched instructions. This design also experimented with different compilation options and their impact on these two metrics.

In the testing process, the selected architecture for this design was RV64G (RV64GC was also used when measuring the number of fetched instructions).

3) Results: **Effectiveness of Standard Subclasses Fusion**

Under O1 optimization, Coremark executed 877,585 instructions before fusion, and after fusion, the number of dynamic instructions decreased to 852,442, resulting in a reduction of 2.87% in dynamic instructions.

Among the eight categories, the Load/Store Pair had the highest occurrence frequency, accounting for 55.13%, followed by Load Effective Address at 36.42%, and Load Immediate Idioms at 4.92%. Indexed Load, Wide Multiply/Divide Remainder, and Post-indexed Memory Operations did not occur. The specific distribution is shown in Figure 26:

Overall Fusion of Standard and Generalized Subclasses

In this section, the design discusses the fusion situation after enabling all standard and generalized subclasses.

Under O1 optimization, Coremark executed 877,585 instructions before fusion of standard and generalized sub-

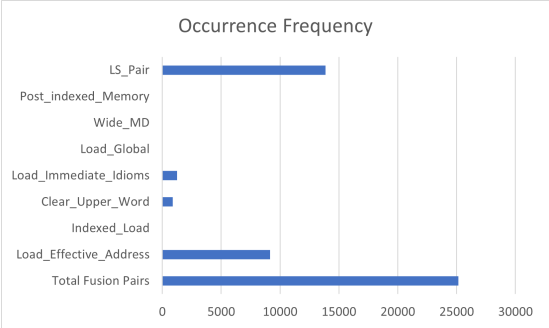

Fig. 26. Occurrence frequency of standard fusion pairs

classes, and after fusion, the number of dynamic instructions decreased to 810,926, resulting in a reduction of 7.60% in dynamic instructions.

Among the eight categories, Indexed Load had the highest occurrence frequency, accounting for 54.59%, followed by Load/Store Pair at 20.83%, and Load Effective Address at 15.83%. Wide Multiply/Divide Remainder did not appear in the instruction sequence.

Comparing the occurrence frequencies of standard subclasses and generalized subclasses, standard subclasses appeared a total of 25,143 times, while generalized subclasses appeared a total of 41,516 times. The proportion of generalized subclasses is much higher than that of standard subclasses, indicating the necessity of adding generalized subclass fusion pairs.

The specific distribution of standard subclasses and generalized subclasses is shown in Figure 27:

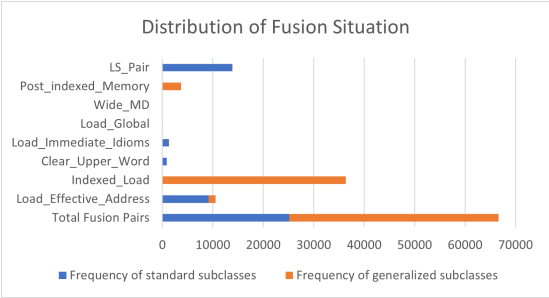

Fig. 27. Fusion situation of standard and generalized subclasses

Cases of Failed Fusion

Cases of fusion failure due to reasons such as register allocation are also important. In the tests, there were 66,659 successfully fused instruction pairs under O1 optimization, while there were 83,651 instruction pairs that matched but failed to fuse due to mismatched register operands, which is much higher than the number of successfully fused pairs. This could be due to the lack of relevant optimization for instruction fusion in the compiler, resulting in improper register allocation, or there may be new opportunities for instruction fusion. Therefore, it is necessary to study the instruction pairs that can actually be fused to adjust the strategies in the compiler or hardware implementation.

The occurrence frequencies of mismatched fusion pairs are shown in Figure 28. Interestingly, both Clear Upper Words and Post-indexed Memory Operations have significantly higher frequencies of mismatch than their fusion frequencies. For example, Clear Upper Words only occurred 883 times before but had a mismatch frequency of 37,498. Therefore, there is still considerable potential for fusion.

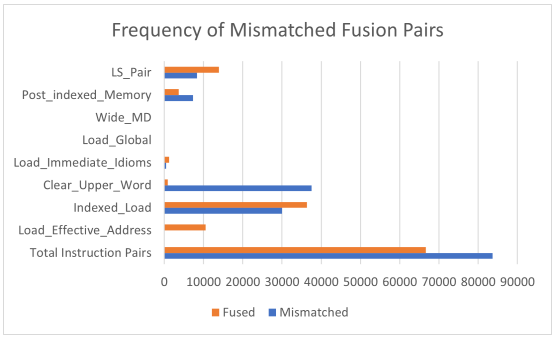

Fig. 28. Occurrence frequency of mismatched fusion pairs

4) Opportunities for Non-Contiguous Fusion: Utilizing fusion opportunities in non-contiguous instructions and memory situations has significant significance, especially for high-frequency fusion pairs. In this design, the non-contiguous situation of load/store pair instructions was specifically studied.

During the fusion process, it is necessary to determine whether there is a data dependency between the region between the head instruction and the tail instruction and the rs register of the tail instruction. This design only considers non-contiguous fusion cases where there is no data dependency.

In the tests, there were 13,882 continuously fused load/store pair instructions under O1 optimization, while there were only 535 non-contiguous fused instructions, accounting for a relatively small proportion. After analysis, it should be noted that the presence of data dependencies is the primary reason for the majority of non-contiguous fusions. Therefore, the focus of future research will be on resolving the data dependency issues in non-contiguous fusion to increase the fusion quantity.

Based on measurements, the average fusion distance was 22.34. This means that it is impractical to simply increase the fetch width in hardware and use combinational logic to directly detect non-contiguous fusion.

Recommended Fusion Scheme

Based on the above analysis, the occurrence frequencies of different subclasses vary. Therefore, this design recommends selecting appropriate additions based on the proportions of each subclass within the total class.

Specifically, this design recommends adding the standard and third subclasses of Load Effective Address, the third subclass of Indexed Load, the standard subclass of Clear Upper Word, the standard subclass of Load Immediate Idioms, the first subclass of Load Global, the third subclass of Post-indexed Memory Operations, and the standard subclass of Load/Store Pair.

Of course, the distribution of various subclasses will vary in different benchmark tests. Load programs should be ana-

lyzed based on the characteristics of the workload to provide guidance for microarchitecture design solutions.

REFERENCES

[1] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In International Conference on Machine Learning, pp. 5338–5348. PMLR, 2020.

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), Advances in Neural Information Processing Systems, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

[3] Oikarinen, Tuomas, and Tsui-Wei Weng. "CLIP-Dissect: Automatic Description of Neuron Representations in Deep Vision Networks."arXiv preprint arXiv:2204.10965(2022).

[4] Eric Wong, Shibani Santurkar, and Aleksander Madry. Leveraging sparse linear layers for debuggable deep networks. In International Conference on Machine Learning, pp. 11205–11216. PMLR, 2021.

[5] Alvin Wan, Lisa Dunlap, Daniel Ho, Jihan Yin, Scott Lee, Henry Jin, Suzanne Petryk, Sarah Adel Bargal, and Joseph E. Gonzalez. Nbdt: Neural-backed decision trees, 2020.

[6] Zhi Chen, Yijie Bei, and Cynthia Rudin. Concept whitening for interpretable image recognition. Nature Machine Intelligence, 2(12):772–782, 2020.

[7] Mateo Espinosa Zarlenga, Pietro Barbiero, Gabriele Ciravegna, Giuseppe Marra, Francesco Giannini, Michelangelo Diligenti, Zohreh Shams, Frederic Precioso, Stefano Melacci, Adrian Weller, et al. Concept embedding models. arXiv preprint arXiv:2209.09056, 2022.

[8] Max Losch, Mario Fritz, and Bernt Schiele. Interpretability beyond classification output: Semantic bottleneck networks, 2019. URL https://arxiv.org/abs/1907.10882.

[9] Mert Yuksekgonul, Maggie Wang, and James Zou. Post-hoc concept bottleneck models. arXiv preprint arXiv:2205.15480, 2022.

[10] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, et al. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In International conference on machine learning, pp. 2668–2677. PMLR, 2018.

[11] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. URL https://arxiv.org/abs/2103.00020.

[12] Bolei Zhou, Yiyou Sun, David Bau, and Antonio Torralba. Interpretable basis decomposition for visual explanation. In Proceedings of the European Conference on Computer Vision (ECCV), pp. 119–134, 2018.

[13] Yu, T., Gao, H., Lin, T. E., Yang, M., Wu, Y., Ma, W., ... & Li, Y. (2023). Speech-Text Dialog Pre-training for Spoken Dialog Understanding with Explicit Cross-Modal Alignment. arXiv preprint arXiv:2305.11579.

[14] Hazarika, D., Zimmermann, R., & Poria, S. (2020, October). Misa: Modality-invariant and-specific representations for multimodal sentiment analysis. In Proceedings of the 28th ACM international conference on multimedia (pp. 1122-1131).

[15] He, J., Yanga, H., Zhang, C., Chen, H., & Xua, Y. (2022). Dynamic invariant-specific representation fusion network for multimodal sentiment analysis. Computational Intelligence and Neuroscience, 2022.

[16] Hu G, Lin T E, Zhao Y, et al. Unimse: Towards unified multimodal sentiment analysis and emotion recognition[J]. arXiv preprint arXiv:2211.11256, 2022.

[17] Chen, F., Luo, Z., Xu, Y., & Ke, D. (2019). Complementary fusion of multi-features and multi-modalities in sentiment analysis. arXiv preprint arXiv:1904.08138.

[18] Yang, K., Xu, H., & Gao, K. (2020, October). Cm-bert: Cross-modal bert for text-audio sentiment analysis. In Proceedings of the 28th ACM international conference on multimedia (pp. 521-528).

[19] Wang, D., Guo, X., Tian, Y., Liu, J., He, L., & Luo, X. (2023). TETFN: A text enhanced transformer fusion network for multimodal sentiment analysis. Pattern Recognition, 136, 109259.

[20] Busso, C., Bulut, M., Lee, C. C., Kazemzadeh, A., Mower, E., Kim, S., ... & Narayanan, S. S. (2008). IEMOCAP: Interactive emotional dyadic motion capture database. Language resources and evaluation, 42, 335-359.

[21] Zadeh, A., Chen, M., Poria, S., Cambria, E., & Morency, L. P. (2017). Tensor fusion network for multimodal sentiment analysis. arXiv preprint arXiv:1707.07250.

[22] Tsai, Y. H. H., Bai, S., Liang, P. P., Kolter, J. Z., Morency, L. P., & Salakhutdinov, R. (2019, July). Multimodal transformer for unaligned multimodal language sequences. In Proceedings of the conference. Association for Computational Linguistics. Meeting (Vol. 2019, p. 6558). NIH Public Access.

[23] Poria, S., Cambria, E., Hazarika, D., Majumder, N., Zadeh, A., & Morency, L. P. (2017, July). Context-dependent sentiment analysis in user-generated videos. In Proceedings of the 55th annual meeting of the association for computational linguistics (volume 1: Long papers) (pp. 873-883).

[24] He, K., Chen, X., Xie, S., Li, Y., Dollár, P., & Girshick, R. (2022). Masked autoencoders are scalable vision learners. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 16000-16009).

[25] Zadeh, A. B., Liang, P. P., Poria, S., Cambria, E., & Morency, L. P. (2018, July). Multimodal language analysis in the wild: Cmu-mosei dataset and interpretable dynamic fusion graph. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. 2236-2246).

[26] Poria, S., Hazarika, D., Majumder, N., Naik, G., Cambria, E., & Mihalcea, R. (2018). Meld: A multimodal multi-party dataset for emotion recognition in conversations. arXiv preprint arXiv:1810.02508.

[27] Majumder, N., Poria, S., Hazarika, D., Mihalcea, R., Gelbukh, A., & Cambria, E. (2019, July). Dialoguernn: An attentive rnn for emotion detection in conversations. In Proceedings of the AAAI conference on artificial intelligence (Vol. 33, No. 01, pp. 6818-6825).

[28] Zadeh, A., Liang, P. P., Mazumder, N., Poria, S., Cambria, E., & Morency, L. P. (2018, April). Memory fusion network for multi-view sequential learning. In Proceedings of the AAAI conference on artificial intelligence (Vol. 32, No. 1).

[29] Morency, L. P., Mihalcea, R., & Doshi, P. (2011, November). Towards multimodal sentiment analysis: Harvesting opinions from the web. In Proceedings of the 13th international conference on multimodal interfaces (pp. 169-176).

[30] Liu, Z., Shen, Y., Lakshminarasimhan, V. B., Liang, P. P., Zadeh, A., & Morency, L. P. (2018). Efficient low-rank multimodal fusion with modality-specific factors. arXiv preprint arXiv:1806.00064.

[31] Soleymani, M., Garcia, D., Jou, B., Schuller, B., Chang, S. F., & Pantic, M. (2017). A survey of multimodal sentiment analysis. Image and Vision Computing, 65, 3-14.

[32] Graves, A., & Graves, A. (2012). Long short-term memory. Supervised sequence labelling with recurrent neural networks, 37-45.

[33] Zadeh, A., Liang, P. P., Poria, S., Vij, P., Cambria, E., & Morency, L. P. (2018, April). Multi-attention recurrent network for human communication comprehension. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 32, No. 1).

[34] Pham, H., Liang, P. P., Manzini, T., Morency, L. P., & Póczos, B. (2019, July). Found in translation: Learning robust joint representations by cyclic translations between modalities. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 33, No. 01, pp. 6892-6899).

[35] Cai, Y., Cai, H., & Wan, X. (2019, July). Multi-modal sarcasm detection in twitter with hierarchical fusion model. In Proceedings of the 57th annual meeting of the association for computational linguistics (pp. 2506-2515).

[36] Truong, Q. T., & Lauw, H. W. (2019, July). Vistanet: Visual aspect attention network for multimodal sentiment analysis. In Proceedings of the AAAI conference on artificial intelligence (Vol. 33, No. 01, pp. 305-312).

[37] Yu, J., & Jiang, J. (2019). Adapting BERT for target-oriented multimodal sentiment classification. IJCAI.

[38] Choi, W. Y., Song, K. Y., & Lee, C. W. (2018, July). Convolutional attention networks for multimodal emotion recognition from speech and text data. In Proceedings of grand challenge and workshop on human multimodal language (Challenge-HML) (pp. 28-34).

[39] Zhao, S., Jia, G., Yang, J., Ding, G., & Keutzer, K. (2021). Emotion recognition from multiple modalities: Fundamentals and methodologies. IEEE Signal Processing Magazine, 38(6), 59-73.

[40] Peng, Z., Lu, Y., Pan, S., & Liu, Y. (2021, June). Efficient speech emotion recognition using multi-scale cnn and attention. In ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 3020-3024). IEEE.

[41] Ling, Y., Yu, J., & Xia, R. (2022). Vision-language pre-training for multimodal aspect-based sentiment analysis. arXiv preprint arXiv:2204.07955.

[42] Liang, Y., Meng, F., Xu, J., Chen, Y., & Zhou, J. (2022). MSCTD: A multimodal sentiment chat translation dataset. arXiv preprint arXiv:2202.13645.
[43] Wu, Y., Zhao, Y., Yang, H., Chen, S., Qin, B., Cao, X., & Zhao, W. (2022). Sentiment word aware multimodal refinement for multimodal sentiment analysis with ASR errors. arXiv preprint arXiv:2203.00257.
[44] Zhao, J., Zhang, T., Hu, J., Liu, Y., Jin, Q., Wang, X., & Li, H. (2022). M3ED: Multi-modal multi-scene multi-label emotional dialogue database. arXiv preprint arXiv:2205.10237.
[45] Zhu, L., Zhu, Z., Zhang, C., Xu, Y., & Kong, X. (2023). Multimodal sentiment analysis based on fusion methods: A survey. Information Fusion, 95, 306-325.
[46] Xiong, Z., Chen, S., Wang, Y., Mou, L., & Zhu, X. X. (2023). GAMUS: A Geometry-aware Multi-modal Semantic Segmentation Benchmark for Remote Sensing Data. arXiv preprint arXiv:2305.14914.
[47] Lai, S., Hu, X., Li, Y., Ren, Z., Liu, Z., & Miao, D. (2023). Shared and Private Information Learning in Multimodal Sentiment Analysis with Deep Modal Alignment and Self-supervised Multi-Task Learning. arXiv preprint arXiv:2305.08473.
[48] Ginés, H. (2018). OpenPose: Real-time multi-person keypoint detection library for body, face, hands, and foot estimation. Retrieved from https://github.com/CMU-Perceptual-Computing-Lab/openpose.
[49] Osokin, D. (2018). Real-time 2d multi-person pose estimation on cpu: Lightweight openpose. arXiv preprint arXiv:1811.12004.
[50] Banbury, C., Zhou, C., Fedorov, I., Matas, R., Thakker, U., Gope, D., ... & Whatmough, P. (2021). Micronets: Neural network architectures for deploying tinyml applications on commodity microcontrollers. Proceedings of Machine Learning and Systems, 3, 517-532.
[51] Nuclei. (2020). Hummingbirdv2 E203 Core and SoC: The Ultra-Low Power RISC-V Core. Retrieved from https://github.com/riscv-mcu/e203_hbirdv2.
[52] LowRISC. (2018). Ariane RISC-V CPU. Retrieved from https://github.com/lowRISC/ariane.
[53] Kung, H. T. (1982). Why systolic architectures?. Computer, 15(1), 37-46.
[54] Hennessy, J. L., & Patterson, D. A. (2011). Computer architecture: a quantitative approach. Elsevier.
[55] OpenXiangshan. (2022). NEMU. Retrieved from https://github.com/OpenXiangShan/NEMU/tree/master.
[56] Waterman, A., Lee, Y., Patterson, D., Asanovic, K., level Isa, V. I. U., Waterman, A., ... & Patterson, D. (2014). The RISC-V instruction set manual. Volume I: User-Level ISA', version, 2.
[57] Gal-On, S., & Levy, M. (2012). Exploring coremark a benchmark maximizing simplicity and efficacy. The Embedded Microprocessor Benchmark Consortium.
[58] Henning, J. L. (2006). SPEC CPU2006 benchmark descriptions. ACM SIGARCH Computer Architecture News, 34(4), 1-17.
[59] Sherwood, T., Perelman, E., Hamerly, G., & Calder, B. (2002). Automatically characterizing large scale program behavior. ACM SIGPLAN Notices, 37(10), 45-57.
[60] Laforest, Charles Eric, et al. "Composing multi-ported memories on FPGAs." ACM Transactions on Reconfigurable Technology and Systems (TRETS) 7.3 (2014): 1-23.
[61] Sharma, H., Park, J., Mahajan, D., Amaro, E., Kim, J. K., Shao, C., ... & Esmaeilzadeh, H. (2016, October). From high-level deep neural models to FPGAs. In 2016 49th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO) (pp. 1-12). IEEE.
[62] Springenberg, J. T., Dosovitskiy, A., Brox, T., & Riedmiller, M. (2014). Striving for simplicity: The all convolutional net. arXiv preprint arXiv:1412.6806.
[63] Guo, Kaiyuan, et al. "A survey of FPGA-based neural network accelerator." arXiv preprint arXiv:1712.08934 (2017).
[64] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4510-4520).