Latent Variable Algorithms
for Multimodal Learning and Sensor Fusion

Lijiang Guo
Department of Intelligent Systems Engineering
Indiana University
Bloomington, IN 47405
`lijguo@indiana.edu`

Abstract

Multimodal learning has been lacking principled ways of combining information from different modalities and learning a low-dimensional manifold of meaningful representations. We study multimodal learning and sensor fusion from a latent variable perspective. In the first part, we present a regularized recurrent attention filter for sensor fusion. This algorithm can dynamically combine information from different types of sensors in a sequential decision making task. Each sensor is bonded with a modular neural network to maximize utility of its own information. A gating modular neural network dynamically generates a set of mixing weights for outputs from sensor networks by balancing utility of all sensors' information. We design a co-learning mechanism to encourage co-adaption and independent learning of each sensor at the same time, and propose a regularization based co-learning method. In the second part, we focus on recovering the manifold of latent representation. We propose a co-learning approach using probabilistic graphical model which imposes a structural prior on the generative model: multimodal variational RNN (MVRNN) model, and derive a variational lower bound for its objective functions. In the third part, we extend the siamese structure to sensor fusion for robust acoustic event detection. We perform experiments to investigate the latent representations that are extracted; works will be done in the following months. Our experiments show that the recurrent attention filter can dynamically combine different sensor inputs according to the information carried in the inputs. We consider MVRNN can identify latent representations that are useful for many downstream tasks such as speech synthesis, activity recognition, and control and planning. Both algorithms are general frameworks which can be applied to other tasks where different types of sensors are jointly used for decision making. To our knowledge, this algorithm is the first that addresses a online multimodal decision making problem.

Contents

1 Executive Summary

We present three multimodal learning and sensor fusion methods from a latent variable perspective. In section 2 we briefly introduce the multimodal learning problem, section 3 reviews background with concrete examples. In section 4, section 5, and section 6 we introduce three different methods: a model-free sensor fusion approach, a model-based sensor fusion approach, and sensor fusion for weakly supervised embedding. In each of these sections we first discuss related works, then present our approach. Experiment results are discussed in section 7. We discuss future works in section 8.

2 Introduction

Multimodal learning has been studied in various forms for decades. Broadly speaking, multimodal input can be considered as streams of loosely synchronized multi-domensional data, with each modality being one subset of dimensions. In simple cases when measurements are both taken in real domain, with certain metrics assumed, we often impose a model which leverages on the dependency across modalities to jointly solve a problem. Two commonly used dependency models are linear correlation and conditional probability.

Consider a example from Shumway and Stoffer [2011] where a patient's biometric markers such as log(white blood count) [WBC], log(platelet) [PLT], and hematocrit [HCT] are used to predict probability of a patient's long term survival using Bernoulli linear regression:

$$\mathbb{E}[Y|X] = f(\beta_0 + X_{\text{WBC}}\beta_1 + X_{\text{PLT}}\beta_2 + X_{\text{HCT}}\beta_3), \tag{1}$$

where f is a proper transformation function. We can consider $X_{\text{WBC}}, X_{\text{PLT}}, X_{\text{HCT}}$ as different input modalities as they are essentially measuring different physical concepts. Classical statistical solutions focus on statistical inference in the sense of quantifying the error in estimating $\mathbb{E}[Y|X]$ using probability distributions. To this end, (1) f is often chosen to be functions which are friendly to manipulate, and (2) the relation between different modalities are chosen to be simple linear additive for the same reason. In this simple linear model for regression, we use linear combinations of the input variables. We can extend this model by considering linear combinations of fixed nonlinear functions of the input variables:

$$\mathbb{E}[Y|x] = f\left(\beta_0 + \phi_1(X_1)\beta_1 + ... + \phi_k(X_K)\beta_K\right). \tag{2}$$

In this model, the relation between different input variables are still linear additive, but we allow for flexible representation, i.e. ϕ, of each variable, i.e. features, to be learned from training by optimizing some criterion.

Different from classical statistics, in machine learning we often face very high dimensional input data, and each input modality has essentially different representation forms; for example, video, audio, and text. The challenging questions in creating intelligent systems for processing audio or video inputs are

1. How to describe the complex relation between a single input with the target output, and

2. How to combine different input modalities when there are no straight forward physical model to describe them.

Let's consider speech recognition as a example. Consider we have two input modalities, audio and video, and text as output. The first question corresponds to how are we going to describe the relation between audio and text, or video and text, respectively. The second question corresponds to how the audio input is related to video input in terms of text generation. In this research, we address the second question in a principled manner, and, equally important, create a multimodal algorithm that can outperform uni-modal algorithms.

We want to focus on multimodal time-series data as they are universal in speech, video, computer network, robot control, etc. Our objective is sequential inference under sequential multimodal input. Consider we have two input streams $\{X_t\}$ and $\{Y_t\}$, and a output steam $\{L_t\}$ where $t \in \{1, .., N\}$ are time indices. Our objective is to predict L_k from $\{(X_t, Y_t)\}_{t=1}^k$. The challenge is to model X_t given $(X_{t-1}, ..., X_1, Y_t, Y_{t-1}, ..., Y_1)$, and to model Y_t given $(Y_{t-1}, ..., Y_1, X_t, X_{t-1}, ..., X_1)$. This is particularly important when one input modality is corrupted, and the multimodal algorithm has to recognize or infer the missing information from other input modalities. In particular, we want a
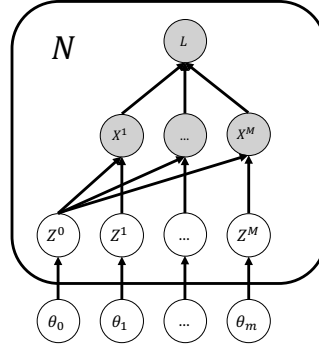
4

Figure 1: Structural generative model.

algorithm which can estimate the reliability of each sensor input. We will discuss this in detail in section 4.

From a generative model perspective, we assume that there are some latent random variable Z which generates the observed random variables X (see Figure 1). In multimodal data, X can be partitioned into disjoint sets of input signal where each $X^{(m)}$ corresponds to one modality. The observed speech activity state L is a function of X. The latent variables Z can also be partitioned into modality invariant and modality dependent subsets $\{Z^{(0)}, Z^{(1)}, ..., Z^{(k)}\}$ where $Z^{(0)}$ is shared by all modality $\{X^{(m)}\}_{m=1}^{k}$, and each $Z^{(m)}$ is specific to $X^{(m)}$ for $m \in \{1, ..., k\}$. Having a shared latent variable $Z^{(0)}$ enables the model to explain for the synchronization between different modalities. In section 5 we discuss this in detail.

We will begin by reviewing a few related topics. Then we present a deep learning based regularized recurrent attention filter, which dynamically combines information from multiple sensor inputs in a sequential decision making task. We present a co-learning mechanism and discuss its advantages in learning robust latent features. Next we marry the strength of deep learning with probablistic graphical models in a coherent way for multimodal learning. We present a multimodal nonlinear state-space model with a structural prior for separating modality dependent and modality invariant features. To demonstrate the proposed algorithms, we tackle the problem of speech activity detection and speech separation in corrupted audio-video streams.

This paper is a first step towards a faraway goal, thus we cannot solve the problem completely. Our model has the following contributions (no. 3, 4 and 5 are unique to our knowledge):

1. The model combines information collected from multiple sensors. Each sensor collects different types of data; for example, image, motion, audio, etc.

2. The model has independent processing module for each sensor. Each module is developed to maximize the utility of its sensor input in a decision making task.

3. The model addresses a sequential decision making task. It (1) takes new inputs at each time step, and (2) generates new attention over modalities by combining the new inputs and its memory, then (3) apply attention to modules to make a final decision for the current time step. The model does not need future information to make decision for current time step.

4. Each module is divided into two sub-components. One sub-component co-learns with other modules; this sub-component allows for simultaneous co-learning features that are shared by all sensors, which allows for robust detection of these features, even a subset of sensors' inputs are corrupted. The other sub-component learns independently from other modules; this allows each module to focus on its sensor's unique input features that are not shared by other sensors. This co-learning design prevents overfitting during training, and improves robustness of the modules.

5. The probablistic approach to co-learning can be used to separate modality-dependent features from modality invariant features. It has wide applications such as speech recognition, speaker identity recognition, etc.

3 Multimodal Learning and Sensor Fusion

3.1 An Example: Audio-Visual Learning

Human senses the world with multiple modalities such as vision, sound, texture, etc. An AI agent, e.g. a robot, also relies on multiple sensors to collect data to perform tasks such as localization, path planning, control a robotic arm, etc. Baltrušaitis et al. [2018] refers a *sensory modality* as our primary channels of communication and sensation, such as vision or touch. A problem is characterized as *multimodal* when it includes multiple sensory modalities. In this work, as a concrete example, we focus on three modalities that are important to human speech: visual, audio, and motion. However, the method we discuss are general enough to tackle other sensor modalities on AI agents.

Many researches leverage natural synchrony between simultaneously recorded visual and audio signals to solve speech related problems. In recent works, lip-reading, the practice of using visual signals to understand speech, has been proven to be able to effectively extract useful information for speech recognition [Ephrat et al., 2017, Assael et al., 2016]. Hence it is natural to consider audio-visual approaches for many speech related tasks, including speech recognition [Ngiam et al., 2011, Mroueh et al., 2015, Feng et al., 2017], speech separation [Hou et al., 2017], and voice activity detection [Ariav et al., 2018].

Among these problems, speech separation is one of the fundamental problems in audio processing. Wang and Chen [2018] gave an overview of recent advances in deep-learning based audio-only speech separation system. Traditionally audio-only methods are used to separate a speech signal from other background signals. When multiple human speeches overlap each other, the speech separation problem is referred to as the *cocktail party problem*. Such a problem is especially challenging to solve because human speeches tend to have similar features than with non-speech sounds. It is also difficult to assign a speech to the corresponding speaker with audio signals only, often referred as *label permutation problem*. A few solutions have been proposed to address multi-speaker speech separation using single channel audio recording. For example, Hershey et al. [2016] proposed a method called deep clustering (DPCL) which uses discriminatively trained speech embeddings to cluster and separate speeches. They also proposed a permutation invariant loss function to solve the label permutation problem. Yu et al. [2017] and Isik et al. [2016] successfully use a permutation invariant loss function to train a DNN for multi-speaker speech separation.

In recent years, deep learning based audio-visual methods have been used for speech separation. Hou et al. [2017] proposed a CNN-based model which outputs a denoised speech spectrogram as well as a reconstruction of the input mouth region. Gabbay et al. [2017] trained a speech enhancement model on videos where other speech samples of the same speaker are used as background noise. Gabbay et al. [2018] use a video-to-sound synthesis method to filter noisy audio. While most audio-visual speech separation methods are speaker dependent, Ephrat et al. [2018] proposed a speaker-independent audio-visual model for separating single speech from a mixture of sounds such as multiple speakers and background noise. They use a pretrained face detection model to identify all speakers in a video, then outputs a complex spectrogram mask for each speaker. Alternatively, Owens and Efros [2018] train a deep neural network to predict whether audio and visual streams are temporally aligned. Learned features extracted from this self-supervised model are then used to condition an on/off screen speakers source separation model. Gao et al. [2018] and Zhao et al. [2018] addressed the closely related problem of separating the sound of multiple on-screen objects (e.g. musical instruments).

3.2 An Example: Load Balancing in Distributed System for Streaming Data

Suppose we have K streams of input data $x^{(1)}, ..., x^{(K)}$, each are different but correlated. They are jointly used to solve a problem whose answer is a stream y. Consider we have many *workers* $\{w^{(1)}, ..., w^{(N)}\}$ who are processing these inputs $\{x_k\}_1^K$. In a distributed system we need to partition the input streams to assign to different workers, and then gather them to solve for the answer. Let's assume for stream k, the information complexity fluctuates in time according to an unknown function $h^{(k)}(x)$, such that the processing time of a worker w is proportional to $t(k, x) = f(h^{(k)}(x))$. For example, a encoded video could take various amount of computations to decode.
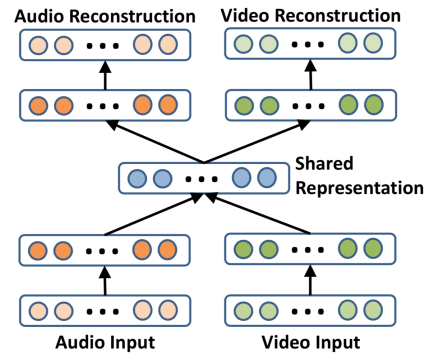
Figure 2: Multimodal deep Boltzmann machine (reproduced from Ngiam et al. [2011]).

3.3 Sensor Fusion

Sensor fusion is one of the core problems in multimodal learning. Technically speaking, in multimodal sensor fusion, we need to integrate information from multiple modalities with the goal of fulfilling some tasks. Comparing with unimodal methods, Baltrušaitis et al. [2018] suggested multimodal sensor fusion has the benefit of (1) providing more robust predictions, (2) allowing the model to utilize complementary information from different modalities, and (3) imputing missing information for corrupted signals.

Sensor fusion algorithms can be divided into two big categories [Baltrušaitis et al., 2018]: model-agnostic and model-based. Model-agnostic sensor fusion does not depend on a specific machine learning method. It usually combines different sensors in a blind way and the relationship between sensors are not exploited. Model-based methods explicitly address fusion in their construction, e.g. using graphical models to impose a structural prior on modalities. In this work, we will propose both a model-agnostic approach and a model-based approach for sensor fusion.

The most common fusion approach is to concatenate the inputs from different sensors at certain modeling stage to solve a unimodal learning problem. Ngiam et al. [2011] proposed a multimodal autoencoder which first use deep Boltzmann machine (DBM) to learn each audio and video features independently, then concatenate the learned features to an multimodal autoencoder to learn a shared representation (Figure 2). To reproduce original inputs, the shared representation is mapped back to each modality. Srivastava and Salakhutdinov [2012] also use DBM to learn a generative model of multimodal input data. Simonyan and Zisserman [2014a] proposed a two-steam convolutional neural networks for action recognition in video. The two-stream model try to capture the complementary information from two modalities: still image and motion between frames. In their work, the last layers concatenates the two streams into a final classification layer. Torfi et al. [2017] proposed a coupled 3D-CNN to map multiple modalities into a representation space to evaluate the correspondence of audio-visual streams. In Ephrat et al. [2018] the audio and visual streams are combined by concatenating the feature maps of each stream, which are subsequently fed into a BLSTM followed by three FC layers.

Whether to combine information at early or late stage affects sensor fusion results. Ngiam et al. [2011] argued that naive concatenation of different sensors' inputs discourages co-learning since the within sensor correlation is much stronger than between sensor correlation, hence the model are biased towards learning dominant patterns in each sensor separately instead of learning patterns that occur simultaneously in multiple sensors. In this regard, they propose to learn joint representations that are shared across multiple modalities at the higher layer of the deep network after learning layers of modality-specific networks. As Sohn et al. [2014] commented, the rationale is that, they believe, the learned features may have less within-modality correlation than raw features, and this makes it easier to capture patterns across data modalities. However, in Ngiam et al. [2011], the middle layer is a simple fully connected layer, and we can only hope that it can always capture meaningful representations shared by all modalities.

One challenge of multimodal learning is how to deal with missing data in different modalities, and infer the missing data from other modalities — it is a unique advantage of multimodal learning

over unimodal learning as we can use the information extracted from one modality to improve the recognition ability of the other modality by complementing the missing information, i.e. given two input streams $\{X\}$ and $\{Y\}$, how to model X_t given $(X_{t-1}, ..., X_1, Y_t, Y_{t-1}, ..., Y_1)$, and Y_t given $(Y_{t-1}, ..., Y_1, X_t, X_{t-1}, ..., X_1)$. A naive approach is to use a autoencoder to map all modalities into a common latent space, and then map back to each modality, hoping that the latent representation can recover all modalities. For example, Ngiam et al. [2011] and Ariav et al. [2018] use a sparse autoencoder to map different modalities into the same space. Ngiam et al. [2011] requires each pre-trained modality-specific network is also jointly trained with other modality-specific networks for cross-modality learning. To infer missing data in one modality, they suppress other modalities with zero input during training.

Model-based sensor fusion explicitly incorporate the relation between sensors into the fusion stage. For example, we can combine multiple sensor signals into modality-dependent and modality-invariant latent random variables through a probabilistic graphical model. Model-based sensor fusion has the advantage of probabilistically detect and recover corrupted signal. We will discuss model-based sensor fusion in detail in section 5.

In a interesting work, Sohn et al. [2014] proposed a multimodal representation learning framework which minimizes the information distance between data modalities through the shared latent representations. To this end, they use variation of information to measure the conditional probability of each modality given the other modalities, thereby determine the quality of shared representation across modalities. As a generative model it can predict missing data modalities given partial observation.

Despite the promise, there still remains missing a principled way of how to learn a good association between multiple data modalities that can effectively deal with missing data modalities in the testing time. Both Ngiam et al. [2011] and Sohn et al. [2014]'s approaches promote learning a shared representation by different modalities, and rely on the collaboration of different modalities. However, co-adaption of different modalities may reduce the individual strength of each modality as we discuss in next sections. We propose two strategies to coordinate the collaboration between modalities. In the first approach we let each local expert focus on it's own modality, and let a dedicated gate expert to coordinate experts. We will discuss this in section 4. In the second approach, we propose a probabilistic graphical model which decompose latent representation into modality dependent and modality invariant features. We will discuss this in section 5.

4 Model Combination for Sensor Fusion

4.1 Motivation

In this section we will develop a model-agnostic sensor fusion algorithm. We are motivated by model combination and aggregation as a way to use parallel models to jointly solve a common problem. Model combination algorithms such as *boosting* has been proven to be a effective approach in reducing prediction error and variance [Bishop et al., 2006]. A natural way to approach multimodal learning is to divide a task into multiple parallel sub-tasks such that each modality uses a dedicated processing module to solve a sub-task. We consider a *conditional mixture model* [Bishop et al., 2006] to coordinate all processing modules towards a common learning goal. From the success of *attention mechanism* in sequence-to-sequence model, we observe that mixing weights can be generated in analogous to a spatial attention through a deep neural network. Therefor we combine model combination methods with recent development in deep learning to propose a recurrent attention filter for multimodal sensor fusion.

4.2 Related Works

4.2.1 Temporal Attention

In sequential prediction problems, sequence-to-sequence (seq2seq) model is a ground-breaking work [Sutskever et al., 2014, Cho et al., 2014a]. One problem with seq2seq networks is that their performance will deteriorate rapidly as the length of input sequence increases [Cho et al., 2014b]. RNN such as LSTM still suffers from difficulty in memorizing long-range relations, especially when a lot of information is cluttered in a sequence. The recently proposed attention mechanism solved this problem by providing a skip-connection and let the decoder focus on a sub-region of the

input [Bahdanau et al., 2014, Chorowski et al., 2015, Mnih et al., 2014, Ba et al., 2014]. Instead of memorizing the entire information of the past sequence, attention filter let the RNN memorize the location of the past information. This greatly reduces the information load of the RNN memory cell. Recently, convolutional and fully-attentional feed-forward architectures like the Transformer model [Vaswani et al., 2017] have emerged as a viable alternative to RNNs for a range of sequence modeling tasks.

In [Bahdanau et al., 2014], attention mechanism was introduced to help the decoder RNN to focus on relevant parts of the input sequence without relying on the RNN to encode all the information through repeated updating the hidden unit. Recall in a seq2seq model, the encoder takes a length M input sequence $\{x_t\}_{t=1}^m$ and generates a *encoder* hidden state sequence $\{h_t^e\}_{t=1}^M$. The last hidden state of encoder h_M^e is used as the initial hidden state for the decoder h_0^d to generate another state sequence $\{h_t^d\}_{t=1}^N$. During training, to making the decoder learn faster, the decoder also takes a auxiliary input which is the true label sequence, $\{y_t\}_{t=1}^N$; during testing, use decoder output prediction y_{t-1} as input to step t. In this simple encoder-decoder structure, suppose the last hidden node of the decoder, h_N^d is related to the first hidden node of the encoder, h_1^e, then the relevant information has to pass through $M + N$ RNN transitions. In general, the average distance between encoder and decoder hidden units is $O(M + N)$ — the number of operations required to relate signals from two arbitrary input or output positions grows linearly in the distance between positions. This long path has made if more difficult to learn dependencies between distant positions. RNN such as LSTM suffers from difficulty in memorizing long-range relations, especially when a lot of information is cluttered in a sequence. To solve this problem, Bahdanau et al. [2014] proposed attention mechanism which provides a skip-connection and let the decoder focus on a sub-region of the input. Instead of memorizing the entire information of the past sequence, attention filter let the RNN remember the location of the past information. This greatly reduce the information load of the RNN. With attention, the average distance between encoder and decoder hidden units is $O(1)$ — a constant number of operations.

The attention mechanism is similar to searching in a database and involves a interaction between query, key, and values. Consider the input to time step t of the decoder as a **query** (i.e. input is the previous hidden state h_{t-1}^d),

$$q_t = h_{t-1}^d. \tag{3}$$

The encoder hidden states $\{h_t^e\}_{t=1}^M$ are **keys** whose information are the **values**. For example, h_t^e is the key to t-th encoder state. To get the t-th decoder output y_t, we search for the relevant encoders values by comparing its key with the query,

$$p(y_t|\{x_t\}_{t=1}^M, \{y_t\}_1^{t-1}) = g(y_{t-1}, h_{t-1}^d, c_t) \tag{4}$$

where c_t (often named the context) is a result of the query procedure. Context vector can be seen as a continuous bag of weighted features of encoder hidden states $\mathbf{h}^e$ [Chan et al., 2016].

Finding the appropriate query procedure is a area of current research. In a simple form, the query is performed in the following steps. First, we compute a similarity measure between q_t and each of h_k^e for $k \in \{1, ..., M\}$, e.g. the kernel inner product $e_{t,k} = \langle q_t, h_k^e \rangle_{W^a} = q_t' W^a h_k^e$ where $'$ indicates transpose and W^a is a (square) matrix. We normalize the similarity over k by

$$w_{t,k} = \frac{\exp(e_{t,k})}{\sum_j \exp(e_{t,j})}. \tag{5}$$

The context is computed as the weighted linear combination of the encoder hidden states

$$c_t = \sum_{k=1}^M w_{t,k} h_k^e. \tag{6}$$

In the original paper that proposed attention, Bahdanau et al. [2014] commented that the approach of taking a weighted sum of all the annotations (e.g. $\{h_k^e\}_{k=1}^M$) is as if computing an expected annotation, i.e. $\mathbb{E}[h^e]$, over all the possible alignments. Here, alignment is defined by the weights $\{w_{t,k}\}_{k=1}^M$, which essentially aligns the t decoder with the entire encoder annotation sequence $\{h_k^e\}_{k=1}^M$. In this sense, $w_{t,k}$ is a estimated probability that the target word y_t is aligned to or translated from a source

342 work x_k. Thus the t-th context vector c_t is the expected annotation over all the annotations with
343 probability $w_{t,k}$.

344 Two kinds of attention models have been proposed. The hard attention model uses the weights to
345 randomly select one location [Mnih et al., 2014, Ba et al., 2014]; the soft attention model uses the
346 weights to form a convex combination of many locations [Bahdanau et al., 2014, Ba et al., 2014].
347 In this work, we choose not to use stochastic "hard" attention for two reasons. First, stochastic
348 attention requires sampling one input stream to make a decision. If both the visual and audio input
349 are noisy, the stochastic attention model has to choose one that is more informative in making a
350 decision. However, if the audio noise and visual noise are independent, the information from each
351 stream may compensate the other. Therefore, we choose to use a deterministic "soft" attention model
352 which dynamically combines multiple streams of data. Second, stochastic attention model involves
353 a intractable objective function with a multinomial latent variable. Monte Carlo and REINFORCE
354 [Williams, 1992] are two common methods for solving this problem. However, it is well known that
355 the both estimation procedures are slow and unstable. On the contrary, soft attention model is fully
356 differentiable and can be optimized using stochastic gradient descent.

357 4.2.2   Spatial-Temporal Attention

358 The temporal attention mechanism enables RNN to selectively focus on a subset of the input sequence.
359 In multimodal learning, in addition to temporal attention, we aim to selectively focus on one (or a few)
360 modalities of current time step and the past, which requires a spatial-temporal attention mechanism.

361 A relevant problem is image caption generation whose task is to transcribe an image into a word
362 sentence that best describes the image. In Xu et al. [2015], a context vector is used to dynamically
363 select relevant parts of a image to generate a word at time t. The context vector is a function of
364 the RNN hidden states of time $t-1$ and the image features. Xu et al. [2015] define a mechanism
365 that generates a positive weight for each image feature location, which can be interpreted as the
366 probability that location is the right place to focus for producing the next word in a sequence.

367 The problem we address here is different in a few places. First of all, in Xu et al. [2015] the image
368 input to the decoder RNN is the same for all time steps. The attention model gives time dependent
369 weight to each location of the image based on the decoder hidden state input and decoder model
370 output of $t-1$. We try to address a problem where at each time step the inputs (e.g. audio, motion,
371 image) is different. This makes our model applicable to, e.g. online video description. Secondly, in
372 Xu et al. [2015] the attention is imposed on the static extracted image features. In our case, because
373 at each time step the inputs are different, we defer attention to each local expert's decision. Third,
374 although the image features are at different locations, they are same type of data and carry similar
375 information. In our model, we have input features of totally different data types, e.g. image, motion
376 and audio. Last, the image features are spatially dependent and the dependency does not change. In
377 video, the spatial dependency of different input streams are dynamic and asynchronous. In summary,
378 we use a dynamic-input cross-modality attention.

379 Another relevant work is dual-stage attention RNN (DA-RNN) [Qin et al., 2017]. Dual-stage attention
380 solves two unique problems different from classical attentions. First, an input attention is used to
381 attentively extract relevant information from multiple parallel exogenous input sequences instead of
382 one input sequence. Then a temporal attention mechanism is used to select relevant encoder hidden
383 states generated by input attention across all time steps. The decoder outputs a time-series prediction
384 instead of a classification. Experiments show that DA-RNN is effective in time-series prediction and
385 robust to noisy inputs.

386 RNN performance will deteriorate rapidly as the length of input sequence increases, Qin et al. [2017]
387 used a temporal attention in the decoder to adaptively select relevant encoder hidden states across all
388 time steps. They use the concatenate score function in [Luong et al., 2015].

$$l_t^i = \mathbf{v}_d^T \tanh(\mathbf{W}_d[\mathbf{d}_{t-1}; \mathbf{s}'_{t-1}] + \mathbf{U}_d\mathbf{h} + i), \quad 1 \le i \le T \tag{7}$$

389 and

$$\beta_t^i = \frac{\exp(l_t^i)}{\sum_{j=1}^{T} \exp(l_t^j)}, \tag{8}$$

10

390   In the encoder, $\mathbf{x}_t \in \mathbb{R}^n$ is the observation of n exogenous driving input sequences at time t. In a
391   normal RNN, the input sequences $\{\mathbf{x}_t\}_{t=1}^T$ is fed to a hidden unit such as LSTM which is updated by

$$\mathbf{h}_t = f(\mathbf{h}_{t-1}, \mathbf{x}_t).$$

392   Qin et al. [2017] proposed to use $\mathbf{h}_{t-1}$ and $\mathbf{x}_t$ to generate a set of n weights for x_t:

$$e_t^k = \mathbf{v}_e^T \tanh(\mathbf{W}_e[\mathbf{h}_{t-1}; \mathbf{s}_{t-1}] + \mathbf{U}_e \mathbf{x}^k) \tag{9}$$

393   and

$$\alpha_t^k = \frac{\exp(e_t^k)}{\sum_{i=1}^n \exp(e_t^i)} \tag{10}$$

394   where $\mathbf{v}_e \in \mathbb{R}^T, \mathbf{W}_e \in \mathbb{R}^{T \times 2m}$ and $\mathbf{U}_e \in \mathbb{R}^{T \times T}$ are parameters to learn. Using the weights α_t, a
395   new input is

$$\tilde{\mathbf{x}}_t = \mathbf{x}_t \odot \alpha_t. \tag{11}$$

396   Then the hidden state at time t is updated as

$$\mathbf{h}_t = f(\mathbf{h}_{t-1}, \tilde{\mathbf{x}}_t). \tag{12}$$

397   With the proposed input attention mechanism, the encoder can selectively focus on certain driving
398   series rather than treating all the input driving series equally.

399   Note in [Qin et al., 2017] the prediction is on $\mathbf{y}_t \in \mathbb{R}$. But in denoising, the prediction is on $\mathbf{y}_t \in \mathbb{R}^D$
400   where D is the number of frequency bins.

4.2.3   Attention mechanism in audio-visual problems

402   Temporal attention mechanism has been used to improve RNN performance for speech related
403   problems. Chan et al. [2016] proposed Listen, Attend and Spell (LAS), which is an audio-only speech
404   recognition model based on sequence-to-sequence learning framework with attention. The model
405   learns to transcribe an audio sequence signal to a word sequence, one character at a time. It consists
406   of an encoder RNN, and a decoder RNN. The encoder RNN is a pyramidal RNN which converts
407   low level speech signals into higher level features. The decoder is an RNN that converts high level
408   features into output utterances by specifying a probability distribution over sequences of characters
409   using the attention mechanism. The encoder and decoder are trained jointly. The key contribution of
410   Chan et al. [2016] is the pyramidal RNN model for the encoder, which reduces the number of time
411   steps that the attention model has to extract relevant information from.

412   RNN has been well known for its power in modeling sequential relation. It is also hard to train and
413   can easily overfit. The attention mechanism has been used as a way to reduce overfitting. Chan
414   et al. [2016] commented that without the attention mechanism, the model overfits the training data
415   significantly and memorizes the training transcripts without paying attention to the acoustics.

416   Chorowski et al. [2015] proposed an attention based audio-video speech recognition model. Their
417   model has a two-stream network with temporal attention in each stream. Each stream's encoder
418   consists of a feature extractor, and a RNN encoder. The RNN encoder takes per-frame input from
419   feature extractor in reverse time order. In the end, the two streams' RNN sequence outputs and final
420   state are fed into a RNN decoder to generate texts. They found the attention mechanism is critical
421   for the speech recognition system to work. Without attention, the model appears to forget the input
422   signal, and produce output sequence that correlates very little to the input. Chorowski et al. [2015]
423   use separate temporal attention to each modality in the decoder. However, the decoder gives equal
424   weights to each modality when generating outputs. That is, each modality receives a sum of attention
425   equals 1. Suppose one modality is corrupted, the weight on that modality is not adjusted. In this
426   regard, Chorowski et al. [2015] does not consider the variation of information in modalities.

427   Here we present a spatial attention based sensor fusion model as a principled way of combining
428   different modality of signals (i.e. audio and video) for e.g. speech enhancement. The model is trained
429   end-to-end, and simultaneously learns spatiotemporal audio-visual features and a sequence model.
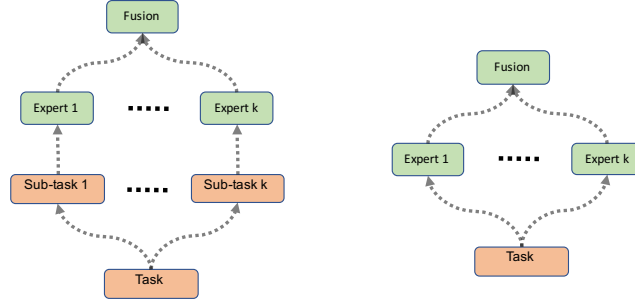
Figure 3: Mixture of sub-task experts (left) and mixture of experts (right). In mixture of experts, each expert receives the identical task. In mixture of sub-task expert, each expert receives a distinct subset of the task.

4.3   Mixture of Sub-task Experts

Dynamically combining multiple functions in supervised learning can be traced back to *mixture of experts* model [Jacobs et al., 1991]. A mixture of experts model is driven by the assumption that a set of training cases may be naturally divided into subsets that correspond to distinct tasks. However, the interference between different subsets of tasks would lead to slow learning and poor generalization. Such interference can be reduced by using a system composed of several different *expert functions* where each one is trained for a subset of tasks. A *gating function* is trained to decide which of the experts should be used for each training case. Instead of a hard decision such as decision tree, the gating function makes probablistics decision by assigning mixing weights to the experts. Neural networks can be used for both the expert functions and gating function, which is known as *mixture density network* (MDN) [Bishop, 1994]. One advantage of MDN is it can approximates a flexible family of distributions, including distribution of multiple modes.

Jacobs et al. [1991] want a system to learn how to allocate cases to experts. They designed a loss function such that the gating network allocates a new case to one or a few experts, and if the output is incorrect, the weight changes are localized to these experts and the gating network. The experts are therefore local in the sense that the weights in one expert are decoupled from the weights in other experts. In addition they will often be local in the sense that each expert will be allocated to only a small local region of the space of possible input vectors.

Different from dividing all training cases into subsets of tasks, we observe that a training case can be divided into *parallel sub-tasks* (Figure 3). For example, in using two hands to open a jar, the parallel sub-tasks are left-hand movement and right-hand movement. In speech, the lip movement and audio are data of two sub-tasks, where speech is the super-task.

Many recent researches have implicitly utilized such a parallel sub-task design. In activity recognition, the two-stream network [Simonyan and Zisserman, 2014a] has a sub-task design for extracting motion and image features. The shared decoder works as a fusion function which takes concatenated inputs from sub-networks. In audio-visual speech recognition, Chorowski et al. [2015] use two encoder neural networks for processing audio and video signals. The two also share one decoder which takes concatenated inputs from the two encoders. In these works, the gating function is implicit in the decoder. Such a blackbox methods does not explicitly assign sub-tasks to experts, and the error backpropogation may cause global changes to expert functions.

We make three extensions to MDN for multimodal sensor fusion. First, we separate expert networks into different functional components by dividing a training case into parallel sub-tasks, one for each sensor. Sub-tasks could have same objective or partially overlapping objectives. Each expert network receives input from only one unique sensor, hence is forced to solve only one sub-task. The gating network receives all sensor inputs to decide a soft mixing weight. This contrasts a boosting algorithm, e.g. mixture of expert, where each expert receives the complete and identical task.

Second, in the mixture of density network, one function (i.e. neural network) is used to predict the parameters of all of the component densities as well as the mixing coefficients, so the hidden units between input and output layers are shared among the input-dependent functions. We further divide

469 the functions into different modular neural networks, such that each expert learns a unique function,
470 e.g. neural network for the assigned sub-task.

471 A major concern of mixture of experts model is co-adaption of experts. The third contribution of
472 our work is to re-address this issue by promoting multiple experts to learn similar features, and also
473 learning dissimilar features at the same time; we propose a mechanism to partition latent variables
474 into co-adaption set and independent set; see subsection 4.5.

4.4   Recurrent Attention Filter for Multimodal Sensor Fusion

476 In this section we introduce recurrent attention filter for sensor fusion. We try to marry the strength
477 of spatial-temporal attention and mixture of sub-task experts. For demonstration, we use speech
478 activity detection where the inputs are audio, image, and motion, and outputs are binary labels. This
479 model can be converted to a, e.g. speech and video enhancement system where outputs are, e.g.
480 spectrograms and images.

481 Let $\{\mathbf{x}_1, ..., \mathbf{x}_T\}$ be a sequence of input signals. Partition $\mathbf{x}_t$ into $\{\mathbf{x}_t^{(1)}, \mathbf{x}_t^{(2)}, \mathbf{x}_t^{(3)}\}$ which are audio,
482 image, and motion features, respectively. These features are functions, e.g. neural networks, of input
483 signals. Let $\mathbf{y} = \{y_1, ..., y_T\}$ denote the sequence of speech activity labels, i.e. $\{0, 1\}$. Let $\mathbf{x}_{\leq t}^{(m)}$ and
484 $\mathbf{y}_{\leq t}$ denote the subsequences $\{\mathbf{x}_1^{(m)}, ..., \mathbf{x}_t^{(m)}\}$ and $\{y_1, ..., y_t\}$. We model y_t as a Bernoulli random
485 variable with conditional distribution:

$$\mathbb{P}(y_t = 1|\mathbf{x}_{\leq t}) = \langle g(\mathbf{x}_{\leq t}), [f_1(\mathbf{x}_{\leq t}^{(1)}), f_2(\mathbf{x}_{\leq t}^{(2)}), f_3(\mathbf{x}_{\leq t}^{(3)})]\rangle, \tag{13}$$

486 where $\langle \cdot, \cdot \rangle$ is Eculidean inner product. Each *expert* function $f_m(\cdot)$ is a neural network which predicts
487 the conditional distribution of y_t given input from modality m only:

$$f_m(\mathbf{x}_{\leq t}^{(m)}) = \mathbb{P}(y_t = 1|\mathbf{x}_{\leq t}^{(m)}). \tag{14}$$

The *spatial attention* function $g : \mathbb{R}^n \mapsto \mathbb{R}^m$ is also a neural network which specifies a mixing
weights for each f_m. Input to g are features of all modalities, as g has to output weights on all
modalities. The output weights are non-negative and sum to 1. Therefore, $y_t|\mathbf{x}_{\leq t}$ has a mixture of
Bernoulli distributions, where each component is a conditional distribution:

$$p(y_t|\mathbf{x}_{\leq t}) = \sum_m p(m|\mathbf{x}_{\leq t})p(y_t|\mathbf{x}_{\leq t}^{(m)}). \tag{15}$$

488 In the *recurrent attention mixture model* (Equation 15), for each expert function we apply temporal
489 attention on the input $\mathbf{x}_{<t}^{(m)}$, such that $f_m(\mathbf{x}_{\leq t}^{(m)})$ can look back in time and focus on relevant time
490 segments. The recurrent attention mixture model can be solved using stochastic gradient descent.
491 RNN with attention is often slow to train and has high memory consumption. Therefore the RNN
492 with temporal attention can be replace with 1-D dilated convolutional neural network.

A simplified version is akin to Markov mixture of expert [Meila and Jordan, 1996] where current
prediction is independent of the past given a latent random variable:

$$p(y_t|\mathbf{x}_{\leq t}) = \sum_m p(m|\mathbf{x}_t, \mathbf{h}'_{t-1})p(y_t|\mathbf{x}_t^{(m)}, \mathbf{h}_{t-1}^{(m)}) \tag{16}$$

493 where $\mathbf{h}'$ and $\mathbf{h}^{(m)}$ are RNN hidden units in attention and expert networks. The *Markov attention*
494 *mixture model* (Equation 16) is shown in Figure 4. Regularization is discussed in subsection 4.5.

495 We can further simplify the model by assuming $\mathbf{y}_t \perp \mathbf{x}_{<t}|\mathbf{x}_t$ such that

$$p(y_t|\mathbf{x}_{\leq t}) = \sum_m p(m|\mathbf{x}_t)p(y_t|\mathbf{x}_t^{(m)}). \tag{17}$$

496 This simplified model (Equation 17) is a *conditional attention mixture model* — each time step
497 is a independently distributed mixture of conditional Bernoulli distributions. We can follow the
498 standard EM receipt [Bishop et al., 2006] to solve the Markov attention mixture model and conditional
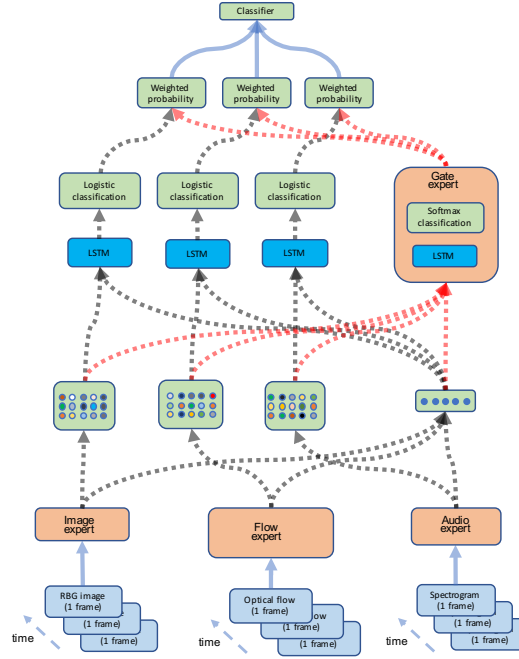499 attention mixture model (see Appendix A).

13

Figure 4: Regularized recurrent attention multimodal sensor fusion model.

4.4.1   Comments

In multimodal sensor fusion, each expert function would process one of the parallel sensor input, rather than a subset of all the training cases as in mixture of experts (Figure 3). The gating function is replaced with a spatial attention function, which at each time step, generates soft attention over sensors.

Baltrušaitis et al. [2018] suggested that late fusion ignores the low level interaction between the modalities. However, Ngiam et al. [2011] found that it is difficult to capture cross-modality relation with low level features, because within-modality correlation is much stronger than between-modality correlation. They suggested that fusion at higher level would encourage learning cross-modality features, because higher level features may have less within-modality correlation than raw features.

In our model, the gating network could take input features from any level of the expert networks. It is possible to apply a hybrid of late and early fusion by taking both high and low level features. Hence our model can be considered as a model agnostic hybrid fusion. These features are used to create dynamic mixing weights, similar to a spatial attention mechanism.

The spatial attention function g is similar to soft attention [Xu et al., 2015]. But there are two key differences. First, $g(t)$ does not have input from $\mathbf{x}_{>t}$; this is critical because g does not rely on future information to make current decision, which allows for online decision making. Second, for $t + 1$, g takes new input $\mathbf{x}_{t+1}$ in addition to previous inputs $\mathbf{x}_{\leq t}$, hence a filter.

The recurrent attention mixture model is similar to mixture density network [Bishop, 1994]. The major difference is that we use dedicated neural networks to predict each component density (i.e. expert) and mixing coefficients (i.e. gate), thus maximizing the utility of each sensor's input. To recognize the covariance between different sensors, we design a co-learning mechanism as explained in next section.

4.5   Co-learning Latent Features

Each expert function f_m is trained to maximize the utility of its sensor input. Although sensor inputs are different, they are likely to have some similar latent features. For example, when uttering some words, part of the lip movements (i.e. visemes) are coordinated with the sound (i.e. phoneme). In this sense, two input modalities and not independent. We can make an assumption that for the speech

528   activity detection task, for each experts, there are some common features that "represent" some states;
529   for example, a fixed set of nodes in both experts' hidden layer are activated when speech is found. If
530   we can apply some prior information which encourage latent variable sharing, we may extract more
531   robust features for the classification task.

532   To this end, we consider two techniques from statistical learning. One is to apply a penalty term to
533   part of hidden unit outputs to encourage the output of different experts be similar to each other. For
534   example, we can add a term to the loss function which is proportional to the sum of squared distances
535   between outputs and average of outputs. Alternatively, instead of tying hidden unit outputs we can
536   put penalty on weights to encourage some weights are tied between different expert networks.

537   A second approach is to define a structure prior of the generative model via probabilistic graphical
538   model. We will discuss this approach extensively in section 5.

4.5.1   Distance based regularization

For each expert network, choose one hidden layer. Let $\mathbf{z}_1, ..., \mathbf{z}_M$ denote the hidden unit outputs of
these layers of the M experts. Let $\mathbf{z}_m^*$ denote the first n hidden nodes of $\mathbf{z}_m$. We assume that

$$\mathbb{E}[\mathbf{z}_m^*] = \zeta. \tag{18}$$

That is, for all input sensors, $\mathbf{z}_m^*$ has the same expectation. Thus

$$\hat{\zeta} = \frac{1}{K} \sum_{k=1}^{K} z_k^* \tag{19}$$

is an unbiased estimator of ζ, where z_k^* is a sample of $\mathbf{z}_k^*$. Define the co-learning loss as

$$\mathcal{L}_{co} = \sum_{m=1}^{M} \lambda_m \|z_m^* - \hat{\zeta}\|_2^2. \tag{20}$$

540   $\mathcal{L}_{co}$ is a sum of squared $L2$ norms; λ_k is a sensor specific penalty parameter selected by cross-
541   validation. Adding $\mathcal{L}_{co}$ to the loss function has the effect of shrinking $\mathbf{z}_k^*$ towards ζ, as in Tikhonov
542   regularization, to prevent overfitting and stabilize parameter estimation.

543   We want to point out that if we assume $\mathbf{z}^*$ are Gaussian random variables, then $\mathcal{L}_{co}$ is equivalent to
544   impose a Gaussian prior distribution over ζ. In this sense, the $\mathcal{L}_{co}$ estimate of $\mathbf{z}^*$ is a MAP estimate.
545   The advantage of formulating $\mathcal{L}_{co}$ as Tikhonov regularization is that we can replace $\hat{\zeta}$ with a moving
546   average of the current mini-batch of samples in stochastic gradient descent.

4.6   Recurrent Attention Filter for Audio-Visual Speech Separation

548   It is easy to modify our model from speech activity detection to speech separation. For speech
549   separation, a common technique is to use a ideal ratio mask (or a ideal binary mask), which is a
550   element-wise ratio (or binarized ratio) between clean and noisy spectrogram. This mask is then
551   multiplied with the input spectrogram to get a denoised spectrogram. The audio input could be
552   either complex spectrogram or magnitude spectrogram. In the case of complex spectrogram, two
553   masks will be generated for real and imaginary component respectively. When using magnitude
554   spectrogram, one mask will be generated, and the phase of noisy input will be used to as denoised
555   phase. The speech activity detection model discussed above can be seamlessly transformed into a
556   speech separation model.

5   Model Based Sensor Fusion: Separating Modality Invariant and Modality Dependent Information

559   In the previous section, we discussed a deep learning model for multimodal sensor fusion. Towards
560   the end, we propose a co-learning idea which encourages co-adaption and independent learning
561   of each sensor at the same time. In the following sections, we propose probablistic models which
562   combines multiple sensors' signals into separate modality-dependent and modality-invariant features.
563   We begin by introduce a variational RNN (VRNN) model where the transition between hidden units

are stochastic. The VRNN model is a non-Markovian model in the sense the conditional independence assumption is broken in both transition and emission. We discuss this property by comparing VRNN with hidden Markov model and linear dynamic system. Using VRNN, we build a multimodal VRNN which imposes a structural prior on the generative model. As a result, the modality dependent and modality invariant factors are encouraged to separate into different latent variables. We emphasis that as oppose to PCA or VAE where latent factors are not known a priori without looking at the posterior distribution, our model explicitly matches latent variables with concepts. This is a result of the explicit graphical model structure.

The non-Markovian model brings a price. One direct outcome is that the latent state no longer contains *all the information* for generation or transition. In this sense, it is not a state-space model in the classical sense. As the latent state does not contain all the information, it is questionable to use the latent state for many downstream tasks; for example, control and planning. Another outcome is that the time derivative information is not captured by the VRNN model as the latent variable only focuses on recovery of the current observation.

5.1 Motivation

Our motivation is driven by a few observations. First, we observe from human representation learning that feature representations are either modality-invariant or modality-dependent: when you listen to a person, your ears hear the speech, and your eyes watch the person's face, then you jointly use the visual and audio signals to decode the content of the speech which is embedded in both audio and visual signals, and other information such as speaker's visual identity and vocal accent which are uniquely embedded in visual or audio signal. With multiple input channels, we can infer the speech content with more precision than with a single input. For this reason, if we consider audio and visual signals are generated by some latent explanatory factors, it is natural to partition the latent factors into modality-dependent and shared subsets.

A second observation is many signals have a temporal structure. If we naively separate each time step as a independent unit for analysis, the data does not carry the same information. A typical approach is to formulate a sequence of latent random variables with Markov property to generate the observed sequence. Two common models are hidden Markov model and linear Gaussian models. However, neither of which are well-suited to modeling long-term dependencies and complex probability distributions over high-dimensional sequences. Neural network models such as RNN have seen many successes in modeling complex sequences with long dependency. One limitation with RNN is structural variations are captured by deterministic transition. While a RNN can increase its memory capacity by increasing the size of its hidden unit, it is more likely to overfit to training data. There is recent evidence that when complex sequences such as speech and music are modeled, the performances of RNNs can be dramatically improved when uncertainty is included in their hidden state. In this work, we combine stochastic latent variable with RNN to take advantage of both methods in a coherent way.

Considering learning a generative model for video of speech. Learning interpretable representations for such data, and comparing them as the speech content or the speaker identity are changed, gives useful high-level tools for speech recognition and speech synthesis. Even though each image is encoded by thousands of pixels and each audio segment is represented as hundreds of frequency bins, the data lie near a low-dimensional nonlinear manifold. A useful model must not only learn this manifold but also provide an intepretable representation of the speech dynamics. A natural representation from speech is that the human speech audio is divided into *phonemes*, and speech video is divided into *visemes* [Bear and Harvey, 2017]. Therefore an appropriate model might switch between discrete states with each state representing the dynamics of a particular action. These two learning tasks, identifying an speech manifold and a structured dynamics model, are complementary. We want to learn the speech manifold in terms of coordinates in which the structured dynamics fit well.

In our preliminary experiments, we used variational autoencoder (VAE) to map images of digit according to content and style by learning from the MNIST dataset. In Figure 5a we project the latent space to a 2 dimension space, and we can see clear clustering of digits, i.e. contents. The row dimension and column dimension show the variation of style of each digit. We also plot the hidden layer activation map in Figure 5b, and we can observe meaningful patterns corresponding to digits.
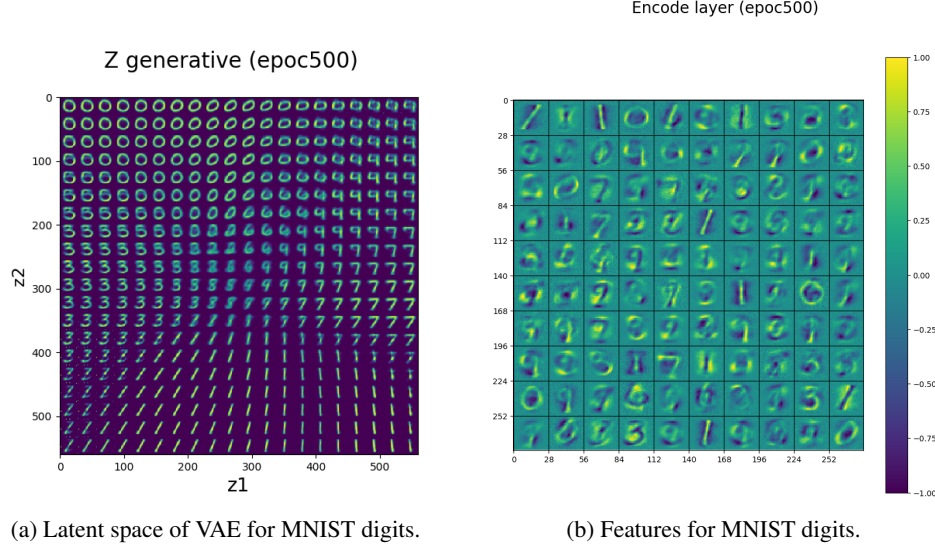
(a) Latent space of VAE for MNIST digits.   (b) Features for MNIST digits.

Figure 5: Separating content and style from MNIST digits.

5.2   Related Works

5.2.1   Inference in Graphical Models with Conjugate Prior

In Gaussian linear dynamic system model with linear Gaussian observation, because the observation model $p(y|x)$ is conjugate to the latent variable model $p(x)$, e.g. Gaussian, then the optimal approximate distribution is naturally Gaussian, and we can use efficient message passing algorithms to perform exact inference. However, when the observation model is not conjugate to the latent variable model, these algorithmic structure break down. A solution is to use general variational inference such as mean-field method. Mean-filed method makes strong independence assumption on the latent distribution for ease of computation, which lead to greater gap between the marginal likelihood and variational lower bound.

5.2.2   Variational Autoencoder

Two common approaches for solving latent variable models are variational inference and Markov Chain Monte Carlo methods. Recently, stochastic variational inference [Hoffman et al., 2013] has been successfully combined with deep neural networks into a class of deep Gaussian models named Variational Autoencoder [Kingma and Welling, 2013, Rezende et al., 2014]. Different from a vanilla autoencoder, a VAE introduces a set of latent random variables $\mathbf{z}$ to explain the variations in the observed variables $\mathbf{x}$. Their joint distribution is defined as:

$$p(\mathbf{x}, \mathbf{z}) = p(\mathbf{x}|\mathbf{z})p(\mathbf{z}). \tag{21}$$

The VAE typically parameterizes $p(\mathbf{x}|\mathbf{z})$ with a highly flexible function approximator such as a neural network. The neural network allows for highly non-linear mapping from $\mathbf{z}$ to $\mathbf{x}$ which is a powerful and unique feature of VAE. However, introducing a highly non-linear mapping from $\mathbf{z}$ to $\mathbf{x}$ results in intractable inference of the posterior $p(\mathbf{z}|\mathbf{x})$. VAE uses a variational approach to approximate the posterior distribution $p(\mathbf{z}|\mathbf{x})$ while incrementally raises the evident lower bound:

$$\log p(\mathbf{x}) \geq \mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \log p(\mathbf{x}|\mathbf{z}) - KL(q(\mathbf{z}|\mathbf{x})|p(\mathbf{z})), \tag{22}$$

where $q(\mathbf{z}|\mathbf{x})$ is a distribution modeled using a inference neural network.

The generative model $p(\mathbf{x}|\mathbf{z})$ and the inference model $p(\mathbf{z}|\mathbf{x})$ are jointed optimized by maximizing the evidence lower bound. The expectation with respect to $q(\mathbf{z}|\mathbf{x})$ is approximated stochastically in VAE.

In VAE we estimate the distribution of latent variables $\mathbf{z}$ and use this information to reconstruct original signal $\mathbf{x}$. Because $\mathbf{z}$ is random rather than deterministic, VAE allows for reconstructing different $\mathbf{x}$ from different modes of $\mathbf{z}$ rather than a single point estimate as in regular auto-encoder.

5.2.3 Recurrent Neural Network

An RNN is a autoregressive model which takes a sequence input $\{\mathbf{x}_t\}_{t=1}^T$ to predict a sequence output $\{\mathbf{y}_t\}_{t=1}^T$. At each time step t, the RNN reads the input $\mathbf{x}_t$ and updates its hidden state $\mathbf{h}_t$ by

$$\mathbf{h}_t = f(\mathbf{x}_t, \mathbf{h}_{t-1}; \theta), \tag{23}$$

where f is a deterministic non-linear function with parameters θ. Common choices of function f can are gated activation functions such as LSTM or GRU.

While the function f is deterministic, we can add randomness to RNN by letting the output be a distribution. For example, assuming a sequence of lag 1 autoregressive relation, i.e. $\mathbf{x}_t$ depends on $\{\mathbf{x}_k\}_{k=1}^{t-1}$. Conceptually, RNN models this sequence by parameterizing a factorization of the joint sequence probability distribution as a product of conditional probabilities such that

$$p(\mathbf{x}_1, ..., \mathbf{x}_T) = \prod_{t=1}^T p(\mathbf{x}_t | \mathbf{x}_{<t}) \tag{24}$$

$$= \prod_{t=1}^T g(\mathbf{h}_{t-1}; \tau) \tag{25}$$

where g is a function which maps the hidden state $\mathbf{h}_{t-1}$ to the conditional probability distribution $p(\mathbf{x}_t | \{\mathbf{x}_k\}_{k=1}^{t-1})$.

We can model the output function g as being composed of two parts. The first part φ_τ is a function that returns the parameter set ϕ_t given the hidden state $\mathbf{h}_{t-1}$

$$\phi_t = \varphi_\tau(\mathbf{h}_{t-1}). \tag{26}$$

The second part of g returns the conditional distribution of $\mathbf{x}_t$ as $p(\mathbf{x}_t | \{\mathbf{x}_k\}_{k=1}^{t-1}) = p(\mathbf{x}_t | \phi_t)$.

Gaussian mixture model (GMM) is a common choice for modeling a high-dimensional and real-valued distribution, especially for modeling structured output density model. For GMM, φ_τ returns a set of parameters including the mixture coefficients and the means and covariance matrices of each Gaussian components.

However, given f is a deterministic function, a RNN model does not have the stochastic transition in a HMM. The lack of a mechanism to model the structural variation imposes a restriction on the RNN, as when it attempts to encode sufficient input variability to capture the signal variations, it inevitably overfits noise variations. In order to prevent overfitting, we must limit the capacity of the RNN and find another mechanism to model signal variations.

5.2.4 Stochastic RNN

Modeling sequential data is a domain of interest to representation learning. In a temporal-aligned RNN, given a deterministic transition function, the only source of variability is the output distribution $p(y_t | x_{\leq t}) = p(y_t | x_t, h_{t-1})$. On the other hand, a limitation of standard HMM is that it is poor at capturing long-range correlation between the observed variables [Bishop et al., 2006]. Recently there is a trend in combining probabilistic models models such as state space model with deep neural networks [Chung et al., 2015]. The key is to incorporate some stochastic hidden states to RNN. Chung et al. [2015] introduced a sequence of latent random variables whose prior distribution at time step t is dependent on all the preceding inputs via a RNN hidden state $\mathbf{h}_{t-1}$. A similar model is proposed by Fraccaro et al. [2016]. These are stochastic RNN models, not state-space models in the strict sense as they have broken the conditional independence assumption in the emission model:

$$p(\mathbf{x}_{1:T} | \mathbf{z}_{q:T}) \neq \prod_{t=1}^T p(\mathbf{x}_t | \mathbf{z}_t). \tag{27}$$

SRNN can be solved using variational inference algorithm. VAE at heart is a simple probablistic graphic model of joint distribution of two time independent random variables $\mathbf{x}$ and $\mathbf{z}$. When there is temporal relation, the assumption is broken. The recognition model would only try to encode the current observation into a latent variable and the generative model would decode the latent variable. Thus VAE would overlook the transition between latent states. In this sense, the VAE model overfits to the training data and discards the temporal information in the training data.
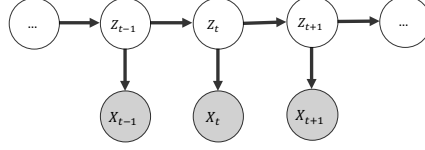
Figure 6: Graphical model of unimodal HMM.

5.2.5 State Space Model: HMM and LDS

A Hidden Markov Model (HMM) (Figure 6) is a doubly embedded random sequence whose underlying Markov chain is not directly observable, hence a hidden sequence. HMM model has three sets of parameters. The first set is the initial state distribution of the Markov chain, $p(\mathbf{z}_0)$. The second set is the transition distribution of the Markov chain, i.e. the conditional distribution $p(\mathbf{z}_t|\mathbf{z}_{t-1})$. The last set is the emission distribution $p(\mathbf{x}_t|\mathbf{z}_t)$. When $\mathbf{z}$ has finite discrete state space, it is a Hidden Markov Model; when $\mathbf{z}$ has continuous state space, it is a dynamical system. For a HMM model, we can choose arbitrary distributions, the efficient forward-backward algorithm can estimate the posterior distribution of the parameters, then use EM algorithm to improve the solution. For a review of HMM see [Rabiner, 1989]. A HMM requires a discrete state space with a known size, a dynamical system extends from finite state space to a continuous state space. A linear dynamical system simplifies a dynamical system by requiring $\mathbb{E}[\mathbf{z}_t|\mathbf{z}_{t-1}]$ to be a linear function of $\mathbf{z}_{t-1}$. Two common forms of LDS are the Kalman filter and Kalman Smoother. To get a linear time algorithm, in a Kalman filter, we take advantage of the conjugacy of exponential family. That is,

$$p(\mathbf{z}_t, \{\mathbf{x}_k\}_{k=0}^t) = p(\mathbf{x}_t|\mathbf{z}_t)\int_{\mathbf{z}_{t-1}} p(\mathbf{z}_t|\mathbf{z}_{t-1})p(\mathbf{z}_{t-1}, \{\mathbf{x}_k\}_{k=0}^{t-1}) \tag{28}$$

which is equivalent to

$$p(\mathbf{z}_t|\{\mathbf{x}_k\}_{k=0}^t) = c_t p(\mathbf{x}_t|\mathbf{z}_t)\int_{\mathbf{z}_{t-1}} p(\mathbf{z}_t|\mathbf{z}_{t-1})p(\mathbf{z}_{t-1}|\{\mathbf{x}_k\}_{k=0}^{t-1}) \tag{29}$$

where $c_t = p(x_t|x_1, ..., x_{t-1})$ is a scaling factor.

The linear Gaussian restriction is that

$$p(\mathbf{z}_t|\mathbf{z}_{t-1}) \sim \mathcal{N}(\mathbf{A}\mathbf{z}_{t-1}, \Gamma) \tag{30}$$
$$p(\mathbf{x}_t|\mathbf{z}_t) \sim \mathcal{N}(\mathbf{C}\mathbf{z}_t, \Sigma). \tag{31}$$

The model can be written in the conventional Kalman filter form as

$$\mathbf{z}_t = \mathbf{A}\mathbf{z}_{t-1} + \mathbf{w}_t \tag{32}$$
$$\mathbf{x}_t = \mathbf{C}\mathbf{z}_t + \mathbf{v}_t \tag{33}$$
$$\mathbf{w}_t \sim \mathcal{N}(0, \Gamma) \tag{34}$$
$$\mathbf{v}_t \sim \mathcal{N}(0, \Sigma). \tag{35}$$

Note here we have omitted the control input $\mathbf{u}$ which is assumed to be known in Kalman filter.

By recognizing the probability density function, we get the posterior distribution $p(\mathbf{z}_t|\{\mathbf{x}_k\}_{k=0}^t)$.

For Kalman smoother, we have the backward form

$$p(\mathbf{x}_t, ..., \mathbf{x}_T|\mathbf{z}_{t-1}) = \int_{\mathbf{z}_t} p(\mathbf{x}_t, ..., \mathbf{x}_T, \mathbf{z}_t|\mathbf{z}_{t-1}) \tag{36}$$

$$= \int_{\mathbf{z}_t} p(\mathbf{z}_t|\mathbf{z}_{t-1})p(\mathbf{x}_t|\mathbf{z}_t)p(\mathbf{x}_{t+1}, ..., \mathbf{x}_T|\mathbf{z}_t). \tag{37}$$

With forward-backward filters, we can get all the statistics we need for inference and learning. For example, we can easily compute the posterior probability of $\mathbf{z}_t$ given the observed sequence (which is used in inference and learning)

$$p(\mathbf{z}_t|\mathbf{x}_1, ..., \mathbf{x}_T) \propto p(\mathbf{z}_t, \mathbf{x}_1, ..., \mathbf{x}_T)$$
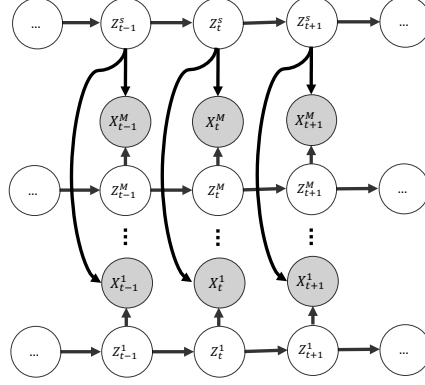
Figure 7: Graphical model of multimodal HMM. Each X^m is a modality for $m \in \{1, ..., M\}$. All modalities have the same shared latent explanatory variable Z^s.

$$= p(\mathbf{z}_t, \mathbf{x}_1, ..., \mathbf{x}_t)p(\mathbf{x}_{t+1}, ..., \mathbf{x}_T|\mathbf{z}_t).$$

In speech recognition, in order to infer which word generated the sound, we need $p(\{\mathbf{x}_t\}_{t=1}^T|W)$ where W is the word decides the probability distribution. With the forward-backwad filter, we have

$$p(\{\mathbf{x}_t\}_{t=1}^T|W) = \int_{\mathbf{z}_t} p(\mathbf{z}_t, \mathbf{x}_1, ..., \mathbf{x}_t)p(\mathbf{x}_{t+1}, ..., \mathbf{x}_T|\mathbf{z}_t) \tag{38}$$

$$= p(\mathbf{z}_T, \mathbf{x}_1, ..., \mathbf{x}_T) \tag{39}$$

where $\mathbf{z}_T$ is the final non-emitting state.

5.3 Multimodal HMM and LDS

First let's consider a multimodal Hidden Markov Model (HMM). Let $\{x_t^m\}_{t=1}^T$ denote the sequence of observed data of modality m, $\{z_t^m\}_{t=1}^T$ denote the sequence of latent explanatory factor of modality m, $\{z_t^s\}_{t=1}^T$ denote the sequence of latent explanatory factor shared by all modalities. The shared latent factor is a key difference between multimodal HMM and regular HMM, as it enables the observed data to be correlated between modalities. When a event (e.g. speech) is described by multiple signal sequences, the sequence of shared latent variables contains modality invariant information such as the semantic content (e.g. speech content), whereas the modality-specific latent variables contain modality-dependent information such as styles pertains only to that modality (e.g. voice timbre, face contour). We define *style* latent variables to be modality-dependent, and *content* latent variables to be modality-invariant. We assume that at every time step t, a modality m has observed variable x_t^m that is generated by latent variables $\{z_t^m, z_t^s\}$. A graphical model of the sequences is shown in Figure 7. The latent variables $\{z_t^s\}$ and $\{z_t^m\}$ have Markov property such that $p(z_k^m|\{z_t^m\}_{t=1}^{k-1}) = p(z_k^m|z_{k-1}^m)$ and $p(z_k^s|\{z_t^s\}_{t=1}^{k-1}) = p(z_k^s|z_{k-1}^s)$. Hence at time t, x_t^m is conditionally independent of the rest of the graph given $\{z_t^m, z_t^s\}$. Notice that the sequence $\{x_t^m\}_{t=1}^T$ is not temporally independent due to temporal dependency in $\{z_t^s\}_{t=1}^T$ and $\{z_t^m\}_{t=1}^T$. $\{x_t^m\}_{t=1}^T$ is also not spatially independent across m at fixed t due to the shared latent variable z_t^s. We assume that modality dependent latent variables are independent, i.e. z_k^m is independent of $z_k^{m'}$ if $m \neq m'$. The assumption of having orthogonal modality dependent latent factors is consistent with the objective to have disentangled latent dimensions [Bengio et al., 2013].

Using the Markov property, we can factorize the joint likelihood in a favorable form. To simplify notation, let's denote all latent variables at time step k by $\mathbf{z}_k = \{\{z_k^m\}_{m=1}^M, z_k^s\}$, and denote all observed variables at time step k by $\mathbf{x}_k = \{x_k^m\}_{m=1}^M$. For a sequence of T steps, the joint distribution of observed variables and latent variables is

$$\mathbb{P}(\mathbf{x}_1, ..., \mathbf{x}_T, \mathbf{z}_1, ..., \mathbf{z}_T) = \prod_{t=1}^T p(\mathbf{x}_t|\mathbf{z}_{\leq t}, \mathbf{x}_{<t})p(\mathbf{z}_t|\mathbf{z}_{<t}, \mathbf{x}_{<t}) \tag{40}$$
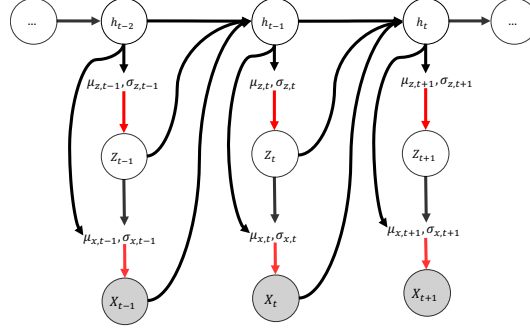
20

Figure 8: Stochastic RNN. Red line is stochastic relation, black line is deterministic relation. h are hidden states of a recurrent neural network, z are latent explanatory variable, x are observed variables, (μ, σ) are parameters of Gaussian distributions.

$$= \prod_{t=1}^{T} \left[\left(\prod_{m=1}^{M} p(x_t^m | z_t^m, z_t^s) \right) \left(p(z_t^s | z_{t-1}^s) \prod_{m=1}^{M} p(z_t^m | z_{t-1}^m) \right) \right] \quad (41)$$

where the second equality follows from Markov property.

Instead of solving this multimodal HMM model, next we will present a multimodal nonlinear dynamical system, and combine it with recurrent neural network, and solve it using stochastic variational inference.

5.4 Multimodal Variational RNN

Recurrent neural networks are able to represent long-term dependencies in sequential data by encoding inputs to update a deterministic hidden state. Recent works found that when complex sequences such as speech and music are modeled, the performance of RNN can be improved by including uncertainty in hidden states [Chung et al., 2015]. The reason is that when there is strong structural variation (i.e. high signal-to-noise ratio), it is mixed with random noise in inputs and outputs. While a deterministic RNN can increase its memory capacity by increasing the size of neural network, it also could bring over-fitting to training data, hence not a good solution.

The first order Markov property in HMM model is an effort to have a compromise between computation complexity and model complexity. However, the marginalization step in finding posterior distribution of z is often intractable. Only a few exact inference algorithms are available (i.e. HMM and Kalman filter).

Having $p(z_t | z_{t-1}) = p(z_t | z_{t<t}, x_{<t})$ would be ideal since it allows for maximum information passing. However, if we consider having a RNN layer on top of the latent variables as shown in Figure 8, the RNN hidden state variable h_t is going to pass all the previous $\mathbf{z}_{\leq t}$ and $\mathbf{x}_{\leq t}$ to timestep t, which breaks down the Markov property. In that case we still can factorize the joint distribution as

$$\mathbb{P}(\mathbf{x}_1, ..., \mathbf{x}_T, \mathbf{z}_1, ..., \mathbf{z}_T) = \prod_{t=1}^{T} p(\mathbf{x}_t | \mathbf{z}_{\leq t}, \mathbf{x}_{<t}) p(\mathbf{z}_t | \mathbf{z}_{<t}, \mathbf{x}_{<t}). \quad (42)$$

This long dependency is challenge to work with in dynamic Bayesian networks. Chung et al. [2015] instead used neural networks to replace the transition probability matrix and emission probability, and solve the estimation problem using stochastic variational Bayes [Kingma and Welling, 2013, Rezende et al., 2014].

In this work, we propose a multimodal stochastic RNN model (Figure 9). In the generative model, there are $M + 1$ RNN sequences $\{\{h^m\}_{m=1}^{M}, h^s\}$, connecting to $M + 1$ latent explanatory variable $\{\{Z^m\}_{m=1}^{M}, Z^s\}$. M of the latent variables $\{Z^m\}_{m=1}^{M}$ are each associated with one modality, and Z^s is shared by all modalities. As we explained earlier, each modality's output X^m is a function of Z^m and Z^s. The RNN are deterministic and the latent explanatory variables are stochastic. By doing so, we let the latent variables model the structural variation in signals, and let the RNN pass information from the past to present. The observed variables stochastically depend on latent variables,
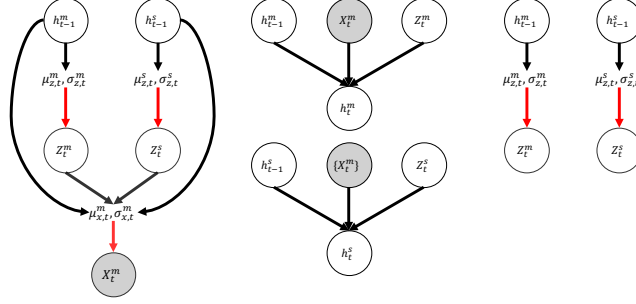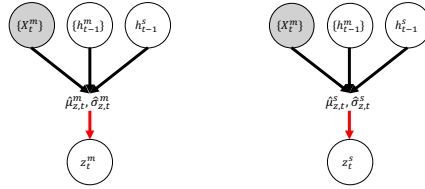
21

Figure 9: Generative model of multimodal stochastic RNN (Figure only shows one modality m). Red line is stochastic relation, black line is deterministic relation. h are hidden states of a recurrent neural network, Z are latent explanatory variable, X are observed variables, (μ, σ) are parameters of Gaussian distributions.

Figure 10: Inference Model of multimodal stochastic RNN (Figure only shows one modality m). Red line is stochastic relation, black line is deterministic relation. h are hidden states of a recurrent neural network, z are latent explanatory variable, x are observed variables, (μ, σ) are parameters of Gaussian distributions.

which allows for noise variations. By have two separate stochastic component, we separate structural variation from random noise variation.

For inference, we consider the following q function as shown in Figure 10 to approximate the posterior distribution $p(\mathbf{z}_1, ..., \mathbf{z}_T | \mathbf{x}_1, ..., \mathbf{x}_T)$

$$
\begin{aligned}
q(\mathbf{z}_1, ..., \mathbf{z}_T) &= \prod_{t=1}^{T} q(\mathbf{z}_t | \mathbf{x}_t, \mathbf{h}_{t-1}) \\
&= \prod_{t=1}^{T} q(\mathbf{z}_t | \mathbf{z}_{<t}, \mathbf{x}_{\leq t}).
\end{aligned} \tag{43}
$$

The last equality holds because in the generative model (Figure 9), $\mathbf{h}_{t-1}$ depends on all the $\mathbf{x}_k$ and $\mathbf{z}_k$ up to time $k = t - 1$. Hence our inference model (Figure 10) implies (43).

Thus we can derive the variational lower bound as

$$
\log p(\{\mathbf{x}_t\}_{t=1}^{T}) \geq \mathbb{E}_{q(\{\mathbf{z}_t\}_{t=1}^{T})} \log \left[\frac{p(\{\mathbf{x}_t, \mathbf{z}_t\}_{t=1}^{T})}{q(\{\mathbf{z}_t\}_{t=1}^{T})} \right] \tag{44}
$$

$$
= \sum_{t=1}^{T} \sum_{m=1}^{M} \mathbb{E}_{q(\{\mathbf{z}_k\}_{k=1}^{t})} \log p(x_t^m | z_{\leq t}^m, z_{\leq t}^s, x_{<t}^m) \tag{45}
$$

$$
- \sum_{t=1}^{T} \sum_{m=1}^{M} KL(q(z_t^m | \mathbf{z}_{<t}, \mathbf{x}_{\leq t}) || p(z_t^m | z_{<t}^m, x_{<t}^m)) \tag{46}
$$

$$
- \sum_{t=1}^{T} \sum_{m=1}^{M} KL(q(z_t^s | \mathbf{z}_{<t}, \mathbf{x}_{\leq t}) || p(z_t^s | z_{<t}^s, \{x_{<t}^m\}_{m=1}^{M})). \tag{47}
$$

Here, the first term is the reconstruction loss. The first set of KL terms is the modality specific latent variable posterior approximation error, and the second set of KL terms is the shared latent variable

22

posterior approximation error. A key difference between RNN-VAE and regular VAE is the the prior distribution for a latent variable Z_t is not the same Gaussian$(0, I)$. Instead, the prior depends on the previous states $Z_{<t}$. Here, we define the prior distribution to be a Gaussian distribution whose parameters are given by a neural network with input h_{t-1} (Figure 9).

6 Learning Latent Variable from Similarity

6.1 Motivation

Measuring the distance between objects has been useful in many clustering or classification tasks. People normally first find a mapping, called a embedding, of a object into a latent space, e.g. a Euclidean space. The mapping is normally hand-crafted. Here we discuss methods which learn a explicit embedding function from only knowing local similarity between sample points, not necessary distance measure, but only label of categories. Such a function can be updated as new data are collected, hence reducing the algorithm complexity. We propose a model which combines embedding and sensor fusion and discuss applications of such approach for tasks such as robust sound event detection and classification.

6.2 Global Embedding from Local Pairwise Similarity

Most methods for learning a latent embedding involve inverting a matrix of pairwise similarity between sample points, e.g. Locally Linear Embedding [Roweis and Saul, 2000] and ISOMAP [Tenenbaum et al., 2000]. In a high-dimensional input space, the cost of inverting such a matrix is $O(n^3)$, and often the matrix is sparse. While there are iterative methods for approximating a solution, there are other methods motivated on a objective function which can be naturally optimized using gradient descent, e.g. *Stochastic Neighborhood Embedding* (SNE) [Hinton and Roweis, 2002] and t-SNE [Van der Maaten and Hinton, 2008]. While LLE and ISOMAP lack a explicit embedding function, SNE and t-SNE models the similarity between two sample points using conditional probability, e.g. Gaussian. Suppose x_i would pick all its neighbors in proportion to their probability density under a Gaussian centered at x_i, the similarity of sample point x_i to x_j is the conditional probability $p_{j|i}$:

$$p_{j|i} = \frac{\exp(-||x_i - x_j||^2/2\sigma_i^2)}{\sum_{k \neq i} \exp(-||x_i - x_k||^2)}. \tag{48}$$

Here σ_i is the variance of the Gaussian that is centered on x_i. In the embedding space (i.e. latent space), a lower dimensional counterpart z_i has a similar conditional probability, which is denoted by $q_{j|i}$:

$$q_{j|i} = \frac{\exp(-||z_i - z_j||^2/2\sigma_i^2)}{\sum_{k \neq i} \exp(-||z_i - z_k||^2)}. \tag{49}$$

If the latent points z_i and z_j correctly model the similarity between the high-dimensional sample points x_i and x_j, the conditional probability $p_{j|i}$ and $q_{j|i}$ will be equal. Motivated by this observation, SNE finds a latent representation which minimizes the difference between $p_{j|i}$ and $q_{j|i}$ for all i, j. SNE minimizes the sum of Kullback-Leibler divergences over all sample points using a gradient descent method:

$$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j_i}}{q_{j|i}}. \tag{50}$$

One of the advantages of t-SNE is when new sample points are available, current embedding can be updated with low cost. For example, for a new point x^*, we can initialize its latent code z^* by the mean of sample latent code, and perform gradient descent for a few iterations.

While SNE is a unsupervised learning approach, its result can be used for classification. After a embedding function f is trained to convergence, it can be applied on new data for tasks involving similarity search. For example, Van der Maaten and Hinton [2008] trained a embedding function of MNIST dataset whose latent space shows good clustering results of digits. A new digit image can be send to the embedding function, and we can perform a k-nearest neighbor search to find its more probable label. Such a search is often cheaper since the embedding space is low-dimensional.

6.3 Weakly Supervised Metric Learning without a Metric

A drawback of SNE and t-SNE is the similarity metric is imposed with a structural form, e.g. Gaussian kernel, on both the input and latent space. Hadsell et al. [2006] proposed a *siamese structure* to learn a globally coherent non-linear function that maps the data evenly in a latent space. The learning relies solely on neighborhood relationships and does not require any distance measure in the input space.

Siamese structure is a weakly supervised model in the sense that the training data only use binary labels to learn a continuous embedding. Recognizing that a meaningful mapping maps similar input vectors to nearby points on the output space and dissimilar vectors to distant points, siamese structure constructs a contractive loss function whose minimization can produce such a embedding function. The loss function takes pairs of samples x_i and x_j, with a binary label $y = 1$ indicating they are similar and $y = 0$ indicating they are dissimilar. In classification, samples from same classes are considered as similar, and from different classes are considered as dissimilar. The output of the function would naturally allocate each sample according to its representativeness in each class without any distance measure. Define the distance function D_W as euclidean distance between the embedding output $G_W(x_1)$ and $G_W(x_2)$ where G_W is the embedding function to be learned:

$$D_W(x_1, x_2) = ||G_W(x_1) - G_W(x_2)||. \tag{51}$$

The contrastive loss function $\mathcal{L}$ is defined as

$$L(W, (x_1, x_2, y)) = (1 - y)L_p(D_W) + yL_n(D_W), \tag{52}$$

where

$$\mathcal{L}(W) = \sum_{i=1}^{N} L(W, (x_1, x_2, y)^{(i)}) \tag{53}$$

where $(x_1, x_2, y)^{(i)}$ is the i-th sample pair, L_p is the loss function for positive pairs and L_n is the loss function for negative pairs. Here L_p and L_n are designed to reflect similarity in the embedding space; for example, Hadsell et al. [2006] suggested using $L_p = D_W(x_1, x_2)^2$ and $L_n = \max\{0, (m - D_W(x_1, x_2))^2\}$ where $m > 0$ is some hyper-parameter represents a margin for dissimilar pares.

Bell and Bala [2015] proposed a *siamese network* which utilizes deep neural network for G_W. The siamese network showed good result in embedding images of objects into a latent space according to their visual similarity represented by class labels.

6.4 A Example: Audio Event Detection in Noisy Environment

Analysis of environmental sound has the potential to be used in many applications, such as surveillance and smart homes. The Detection and Classification of Acoustic Scene and Events (DCASE) challenge is a venue for researchers to propose new methods for audio classification. Several tasks has been defined for audio classification including acoustic scene classification, sound event detection, and audio tagging. Recently, Google released an ontology and human-labeled large scale data set for audio events, i.e. Audio Set [Gemmeke et al., 2017], which consists of 527 classes and over 2 million human-labeled 10-second long sound clips drawn from YouTube videos. Audio Set is defined for tasks such as audio tagging. The objective of audio tagging is to perform multi-label classification on fixed-length audio chunks without predicting the precise boundaries of acoustic events. Recently, deep learning methods have been successfully applied to audio tagging.

Siamese network has been used for sound event detection [Zhang and Duan, 2016, 2017]. Here we explore sound event detection in a noisy environment by combining denoising autoencoder, siamese network, and conditional attention mixture model (Figure 11). We use siamese network to train a embedding encoder for a map of clean sound categories in e.g. Audio Set based on category label. Because we are not performing speech separation, we do not have to stick to STFT features as input to the network. Instead, we use pretrained features such as Audio Set VGGish features which are proved to be more effective than STFT features [Hershey et al., 2017]. We also compare other audio features such as mel-frequency cepstral coefficient (MFCC) features and SoundNet features [Aytar et al., 2016]. We extract VGGish features from audios for downstream tasks as described below.

In a noisy environment, the clean sound is mixed with noise, such that the trained siamese encoder would not embed noisy sound to the exact embedding location of the clean sound. To this end, we
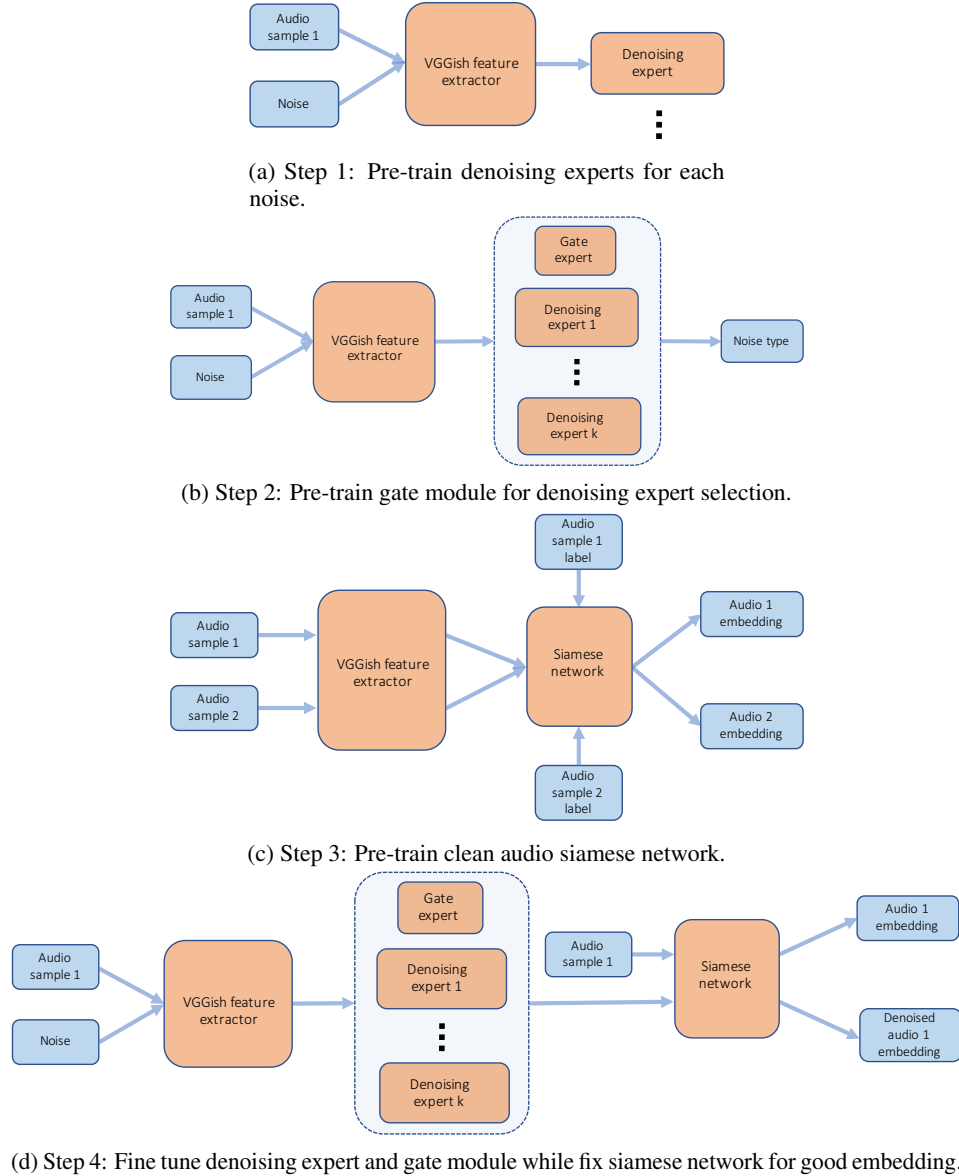
(a) Step 1: Pre-train denoising experts for each noise.

(b) Step 2: Pre-train gate module for denoising expert selection.

(c) Step 3: Pre-train clean audio siamese network.

(d) Step 4: Fine tune denoising expert and gate module while fix siamese network for good embedding.

Figure 11: Robust sound event detection model.

train a denoising autoencoder (DAE) [Vincent et al., 2010] such that the denoised sound when passed through the trained siamese encoder would map to the original clean sound embedding location as close as possible. That is, the output of the DAE is connected to the input of the siamese network to get a embedding. The loss is the difference between clean-sound embedding and denoised-sound embedding. During back-propogation, we can (1) only update the weights for the denoising network, while fixing the siamese network, or (2) only update the weights for the siamese network, while fixing the denoising network, or (3) update both. Using the siamese embedding of the denoised sound, we can classify the class of the sound event by nearest neighbor search using euclidean distance in the embedding space. This has the advantage of searching by *audio similarity* [Zhang and Duan, 2017].

Because noise type can vary, we apply conditional attention mixture model for dynamic selection of DAE. The system is developed in four stages (Figure 11). We first pick a few kinds of noises, then pre-train one DAE for each of them (Figure 11a). On top of pre-trained DAE's, we add the conditional attention mixture model (Equation 17) where each DAE is a expert network, and let the gate module dynamically decide which kind of noise is most likely. This is supervised training since we know which DAE is correct during training time (Figure 11b). Meanwhile in parallel, we train a
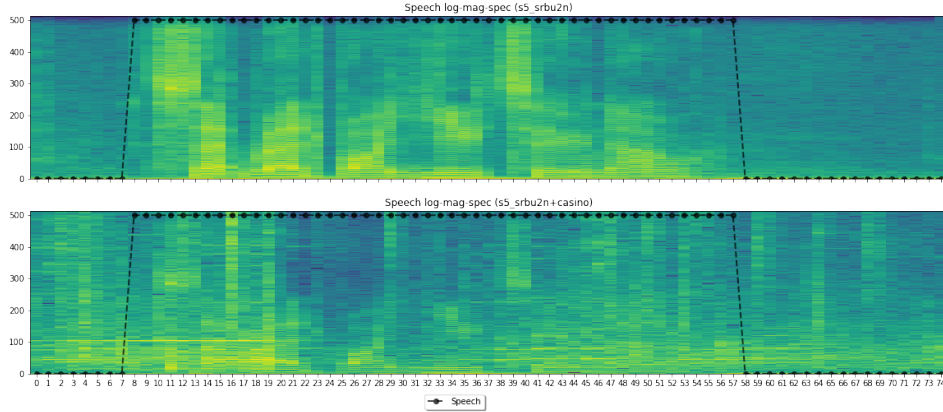
Figure 12: Log-mag-STFT of speech. Top figure shows speech, bottom figure shows speech plus casino noise with SNR = -5DB. Dotted black line shows speech activity.

siamese network on clean audio (Figure 11c). After both the gate module and siamese network are trained, we connect the denoising network with the siamese network (Figure 11d). We further fine tune the DAE's based on the distance of embedded pairs while holding siamese network as ground truth mapping. Recall the conditional attention mixture model applies a probability to each expert module, hence the choice of denoising network could be combined, and consequently the embedding into latent space could be a set of points of different weights. During test time, we can use a weighted mean of these points as the center for nearest neighbor search.

7   Experiments

We will first give a brief overview of speech activity detection and speech separation. After that we will introduce the experiment dataset and video and audio preprocessing protocol. We discuss some relevant existing deep learning techniques that can be used to improve model performance, e.g. pre-trained feature extractor. Then we explain the experiment set-up and results. Finally we discuss the experiment results and propose follow-up works to be completed in our follow up works. Due to time constraints, we are not able to complete all the works we hoped at this time. We plan to complete them as part of the dissertation thesis.

7.1   Speech Activity Detection and Speech Separation

Acoustic event detection (AED) has draw many attentions due to its wide application in intelligent systems such as robots and smart homes [Gemmeke et al., 2017, Hershey et al., 2017]. With new tools such as deep learning, we have seen significant improvement in the results in recent DCASE Acoustic Scene Classification (ADC) task. As one kind of AED tasks, Speech activity detection (SAD) is a classification problem of a given sequence of audio frames into speech active and non-active states. Most SAD models rely on audio signal, for example Li et al. [2017] used grid LSTM to detect speech endpoints. Although speech is audio signal, video signal has shown value to detection and understanding of speech. Sadly, most SAD systems developed so far either entirely rely on audio or video alone, only a limited number of systems utilize both audio and video signal, e.g. [Ariav et al., 2018]. In addition, it is not a straightforward task to effectively combine audio and video signals due to the natural differences between audio and video signals.

A unimodal system that relies solely on the audio signals can fail to do this job due to the common artifacts found in the real-world speech signals such as additive noise and reverberation. To give a concrete example, one speech audio is displayed as log transformed magnitude Short Time Fourier Transform (log-mag-STFT) in Figure 12. Notice although the speech only appears between 8th frame and 57th frame, it is difficult to determine where are true speech activities due to the noise we manually injected. In this scenario, video provides a opportunity to improve the accuracy of speech activity detection.
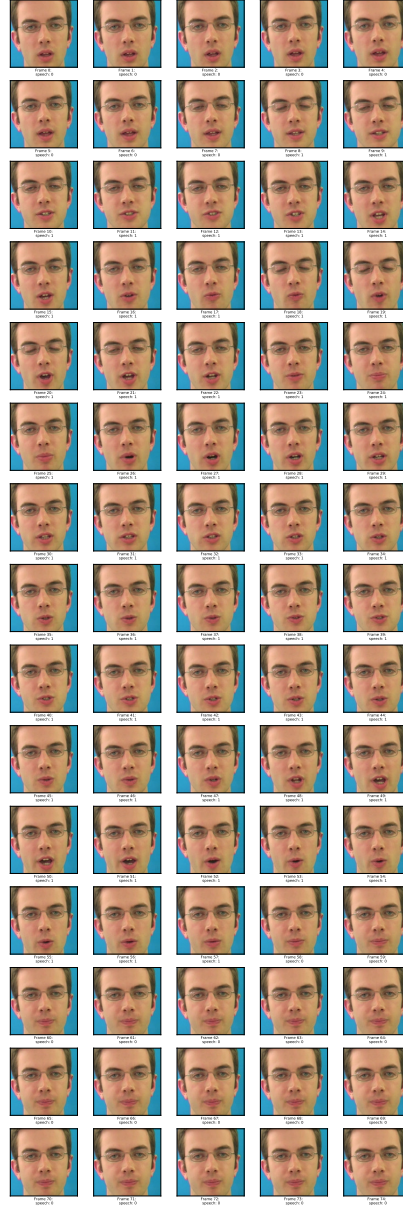
Figure 13: Images of speech (3 seconds, 25fps). Speech activity is from frame 8 to frame 57.

While video signal is invariant to acoustic environments, the relation between speech and face movement is dynamic and usually asynchronous. For example, when uttering some words, part of the lip movements (i.e. visemes) are loosely coordinated with the sound (i.e. phoneme). In this sense, two input modalities are not independent. Visually speaking, speech is closely related to mouth movements. Hence the visual speech activity detection problem can be easily contaminated by non-speech mouth movement such as breathing, eating, etc. For example, in Figure 13, speech starts at 8th frame, but there is noticeable mouth movement from frame 0 to 7. Phonetically speaking, vowel is a speech sound tends to require relatively open mouth, while a consonant is a sound made with mouth relatively closed. Hence speech does not necessarily activates expressive mouth movements, which puts challenge on video-only speech activity detection. More importantly, visual signals do not always provide a reliable cue due to variations such as head rotations, illuminations, different view points, etc.

We want to comment that video based Automatic Speech Recognition (ASR) has been addressed in a few previous researches; for example Assael et al. [2016]. These models directly classify video segments into words or phonemes, and hence can be used for SAD. We approach the SAD problem from a different perspective. We consider ASR as a multi-stage system, and SAD is a key step in front end processing. Instead of using video to predict text, we use visual cue to assist audio cue to classify noise audio from noisy speech audio. When noisy speech audio can be located (e.g. black line in Figure 12), we can apply source separation methods to extract clean speech from noisy speech. The denoised speech is subsequently fed into a audio based ASR system. To this end, video based ASR can work with audio based ASR in our model jointly in a multimodal ASR task.

A SAD system can be applied to convert a unsupervised speech separation systems into a supervised one. Consider a noisy acoustic environment, if the type of noise is unknown, we have a unsupervised speech separation problem. If we know the type of noise, we have a supervised speech separation problem, which is significantly less challenging than the unsupervised case. With a SAD system, we can detect the noisy period immediately before speech. If we assume that the same type of noise will continue during speech, then we can use dictionary based speech separation algorithms such as [Smaragdis et al., 2007] and [Guo and Kim, 2018] with a known dictionary, hence a supervised speech separation system. In a multi-person speech separation scenario, SAD system can assist a face recognition algorithm to determine which person is speaking, and use beam-forming to enhance that speaker's speech.

We emphasize again it is easy to modify our model for speech activity detection to speech separation. For speech separation, a common technique is to use a ideal ratio mask (or a ideal binary mask), which is a element-wise ratio (or binarized ratio) between clean and noisy spectrogram. This mask is then multiplied with the input spectrogram to get a denoised spectrogram. The audio input could be either complex spectrogram or magnitude spectrogram. In the case of complex spectrogram, two masks will be generated for real and imaginary component respectively. When using magnitude spectrogram, one mask will be generated, and the phase of noisy input will be used to as denoised phase. The speech activity detection model discussed above can be seamlessly transformed into a speech separation model.

7.2 Data

In this experiment we use the GRID corpus data [Cooke et al., 2006] which contains 34 speakers. Each speaker has 1000 short speeches, each about 3 seconds long, recorded in a controlled environment with limited ambient noise. The speech is fully annotated, with phonemes mapped to time. Figure 14 shows a sample image of one video.

Due to limitation in computing resources, in this pilot study we randomly choose a subset of subjects to develop our model. For the experiment results presented in the following sections, we use 4 subjects, 2 males and 2 females, to train and test our model. For each subject, we randomly partition the video clips by $7 : 2 : 1$ into training, validation, and in-sample testing sets. We have a 5th subject served as out-of-sample testing set. We are planning on extend to all subjects in following works.

7.3 Video Pre-processing and Pre-training

Each video clip is 3 seconds long with 25 fps. We pre-process the video data using `opencv-python`. However, due to `ffmpeg` issue, some unpacked videos have less than 75 frames. To align the video

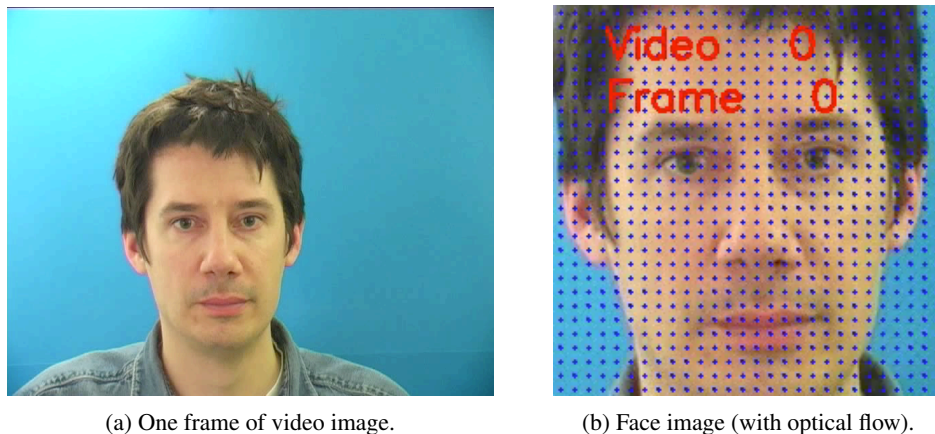(a) One frame of video image.   (b) Face image (with optical flow).

Figure 14: Video image of GRID.

with audio, we pad the first and last frames to the front and end of video to increase the number of frames to 75.

In visual activity detection, still image and motion between frames have been shown to complement each other [Simonyan and Zisserman, 2014a]. Therefore, we extract two kinds of information from video: image and motion. We first extract a face from each video frame. Because the video is relatively clean with speaker's front face at the center the video, we found a Viola-Jones type face detector works well. We used `opencv-python` Harr cascade front face feature to detect faces. We are able to detect high quality face bounding box because the camera is directly pointing at face of the subject, and both camera and subject are stationary. However, there is still movements of subjects between frames, such that each bounding box shifts a little. Because the subject in each video is always stationary and the movement is not dramatic, we choose to use the bounding box in first frame for the 75 frames. Each face image is resized to 224×224 pixels in order to feed into a InceptionV3 feature extraction.

To create image features, pre-trained feature extractors can be used such as AlexNet [Krizhevsky et al., 2012], VGG [Simonyan and Zisserman, 2014b], InceptionV3 [Szegedy et al., 2016] and ResNet-50 [He et al., 2016]. In this experiment, we use InceptionV3 pre-trained on ImageNet without fine-tuning. The extracted InceptionV3 feature has dimension of 2048 for each $224 \times 224 \times 3$ image input. Alternatively, with enough data and computing resource, we could also train the feature extractor from scratch on GRID images or a general human face dataset, and fine-tune with the rest of the model. We will pursue these works in future.

Convolutional network trained on multi-frame dense optical flow has been shown to achieve good performance on capturing human motion [Simonyan and Zisserman, 2014a]. We compute dense optical flow using Farnback algorithm provided in `opencv-python` from the extracted face images. Optical flow is $224 \times 224 \times 2$ for each frame.

To create a corrupted video signal, we randomly apply a square masking patch, and randomly adjust the brightness of that patch area. The location of the patch is random for each video, and the size of the patch is randomly chosen between 50 and 150. For each video, we randomly pick a segment of 20 to 30 frames, and apply the patch to all frames in that segment. This mimics the effect of having a shadow or spot-light casting on the subject's face during a segment of a video clip. Figure 15 shows a example of corrupted image and corrupted optical flow of the same video.

7.4   Audio Pre-processing and Pre-training

For audio data, we first pre-process the input audio to synchronize with images extracted from video by matching number of audio frames with the number of image frames. We choose the STFT window size and re-sample audio such that the there are 25 frames per second for audio. We first down-sampled the speech audio from 50kHz to 16kHz, then perform Short-Time-Fourier-Transform with a window size of 1024 and a stride of 760 which transforms each audio into 75×513 representation.
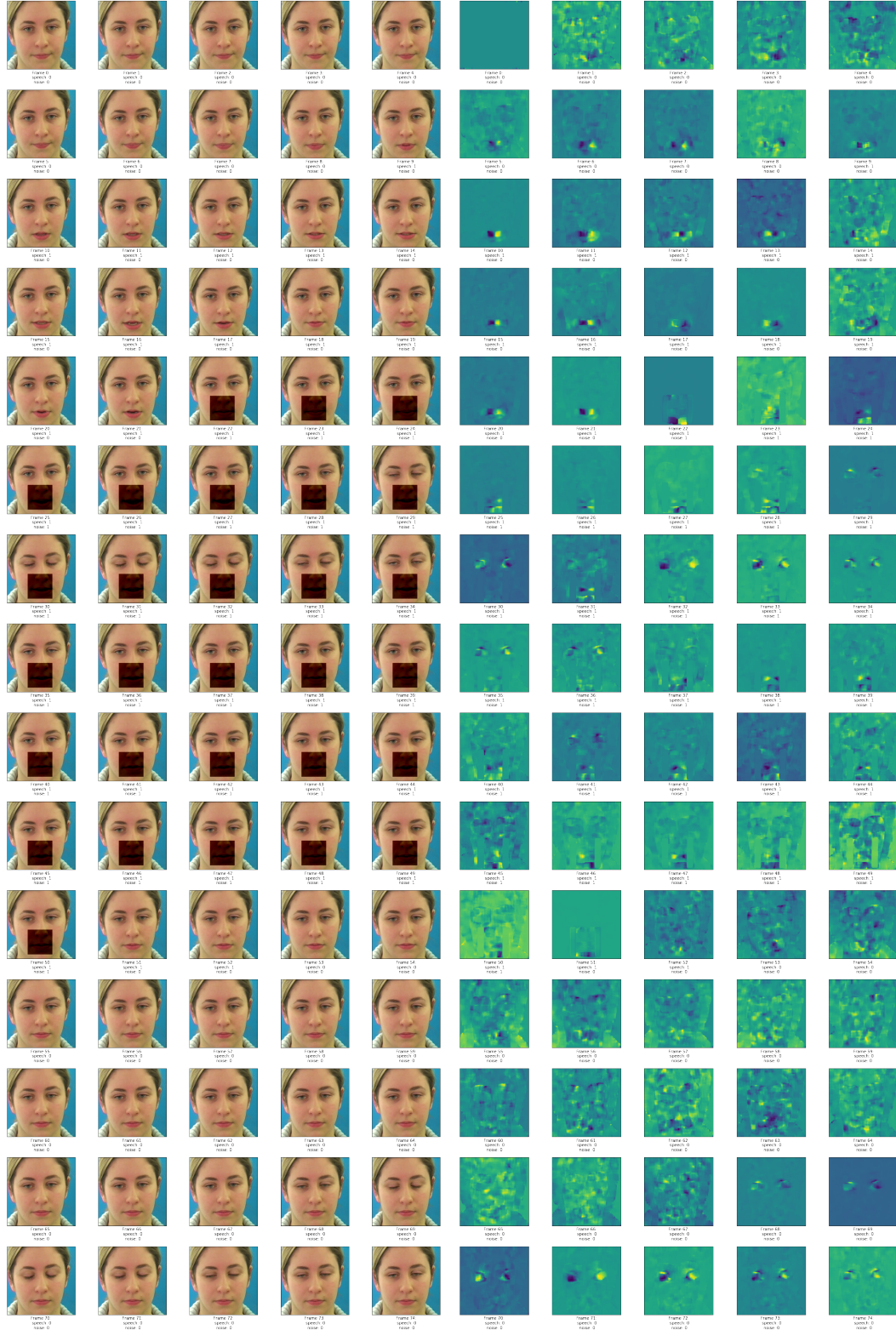
Figure 15: Corrupted images and optical of speech (subject 11, speech srbt7s, 3 seconds, 25fps). Speech activity between 9th and 52nd frames, video noise between 22nd and 49th frames.
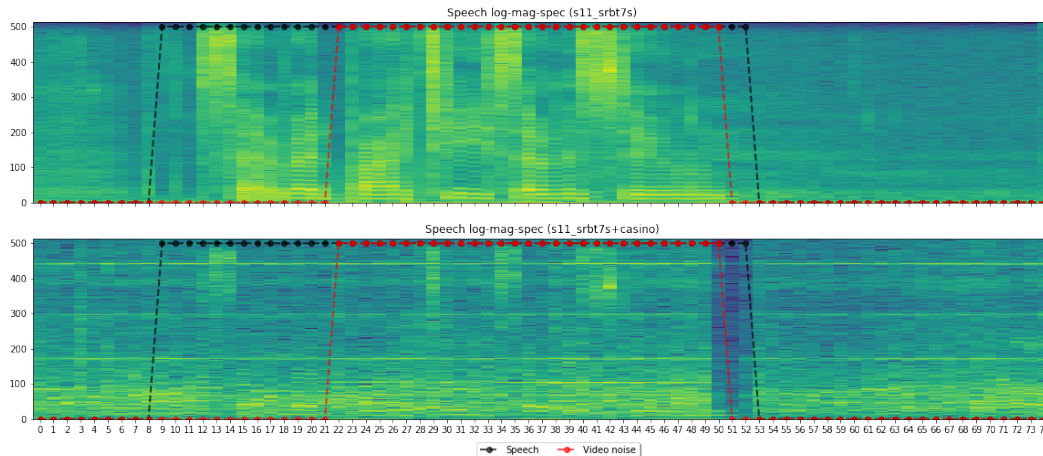
Figure 16: A example of corrupted speech. Top clean speech, bottom corrupted speech (subject 11, speech srbt7s, -5db SNR with Casino noise). Black line marks speech activity, red line marks visual noise (audio noise is presented in all frames).

Notice that 25 frames per second may not be sufficient to fully capture the dynamics of speech. Our method can be easily extended to extract 50 or 75 audio frames per second.

Similar to image feature extractor, we consider audio features such as mel-frequency cepstral coefficient (MFCC) features, SoundNet features [Aytar et al., 2016] and Audio Set VGGish [Hershey et al., 2017]. Because MFCC and VGGish features are not invertible, in speech separation, we use STFT features as input to the network. In acoustic event detection, MFCC, SoundNet and VGGish are proved to be more effective than STFT features [Hershey et al., 2017]. We will pursue these works in future.

To create a corrupted speech signal, we injected two kinds of noise to the audio. The first kind is a casino noise proposed by [Duan et al., 2012]. The casino noise is challenging due to being a non-stationary, wide-band noise which span from low to high frequency domain. Figure 16 shows a example.

We construct a second kind of noise using a mixture of three person from TIMIT training data (TIMIT-3P). TIMIT training set contains 136 female and 326 male speakers, while the testing set contains 56 female and 112 male speakers, which are from eight dialect regions in the US. Each TIMIT speaker has 10 short utterances. For each GRID speech, we randomly choose 3 speeches from TIMIT, then choose a random segment of 3 seconds from each chosen speech. We mix the 3 speeches with equal magnitude. Finally, we mix the 3-person mixture with GRID speech with SNR equal 0 DB.

7.5   Training

During each training epoc, a minibatch of size 16 is used. Same minibatch size is used for evaluating validation and testing set. For RNN we used a 5 frames for truncated backpropagation through time. For DNN model, we feed 5 audio frames per sample to provide a context, which is a common practice in audio based speech recognition system. All models are trained on a server with 4 Nvidia Titan X GPUs with 12GB RAM per GPU. The multimodal RNN model takes about 1-4 hours [1] per training epoc, and we trained it for three days of 30 epocs. The recurrent attention mixture model takes about 4 hours per epoc to train .

7.6   Experiment Results

Our first set of experiments is on the recurrent attention filter. We implemented four versions of the model. The first one is the conditional attention mixture model (Equation 17). The second one is the

---

[1]It seems the training speed is fluctuating on the server for some unknown reason.

Table 1: Experiment Results

	Train(*)	Test (+)
Uni-modal input		
Image	0.12	80.8
Optical flow	0.03	90.5
Audio (with casino noise)	0.06	92.5
Audio (with 3 persons noise)	0.06	91.1
Multimodal input without co-learning		
Conditional Attention Mixture (with casino noise)	0.03	96.38
Conditional Attention Mixture (with TIMIT-3P noise)	0.04	96.25
Recurrent Attention Mixture (with casino noise)	0.03	93.75
Recurrent Attention Mixture (with TIMIT-3P noise)	0.1	85.7
Multimodal input with distance based co-learning		
Conditional Attention Mixture (with casino noise)	0.03	91.74
Conditional Attention Mixture (with TIMIT-3P noise)	0.1	84.56
Recurrent Attention Mixture (with casino noise)	0.03	89.37
Recurrent Attention Mixture (with TIMIT-3P noise)	0.1	76.23
*: MSE, +: % correct.		

recurrent attention mixture model (Equation 15). For each of these two models, we construct two versions, i.e. with and without co-learning (distance based regularizer). For comparison, we also implemented three uni-modal DNN models (audio, optical flow, and image). The results are shown in Table 1.

First we notice that multimodal input model does outperform unimodal input model. While the no-audio models (i.e. image and optical flow) have lower accuracies than the audio-only model, most multimodal input models outperform audio-only model, which suggests image and motion provides addition information not seen in audio for speech activity detection. This validates our hypothesis that multimodal input is useful to speech related tasks.

Second, we notice that the RNN model easily overfits the training data. While the recurrent attention model has lower training error than the conditional attention model, during test time the result is reversed. This suggests we may need (1) to regularize RNN such using e.g. dropout or (2) stop RNN training earlier.

Third, we notice that a simple distance based regularizer for co-learning does not improve prediction accuracy. On the contrary, the un-regularized models have higher prediction accuracy, very likely due to the lack of restriction of the model. Recall our motivation for the model-based regularization is that co-learning helps to separate modality invariant features from modality-dependent features, however, such features may not be useful for speech activity detection. This suggests we may need to carefully design the co-learning structure in order to extract features that are useful to a task.

7.7   Spatial Attention Outputs

We would like to discuss the spatial attention generated by the gate module. Recall the spatial attention gives the mixing weights to the experts. When a sensor's signal is corrupted, we expect the mixing weight for that sensor would decrease, and the mixing weight for a robust sensor would increase, such that the overall decision is more likely to be correct. In Figure 18 we show the attention weights for a speech sample. Notice that speech begins at the 9th frame (black line), and the model prediction lags only one frame. The interesting phenomenon is that at the 22nd frame, the video stream changes to corrupted (red line), causing the mixing weight for audio expert to increase, while the mixing weights for image and optical flow decrease. This suggests that the attention module has learned to identify which sensor inputs are more reliable and assigns mixing weights accordingly. At the 51st frame, the video stream changes to uncorrupted, and we see the mixing weight for image increases to the largest. Recall we have overlap speech audio with noise audio of the entire speech duration, hence video signal is more reliable when uncorrupted.

We notice throughout the entire speech, the attention module prefers to choose only one of the three inputs as the dominant input. It is likely due to the nature of the relationship between the three inputs we have in this particular experiment. Another possibility is the softmax function we used for generating mixing weights may prefer sparse outputs. In order to investigate this property, we can (1) use a different set of sensor inputs, and (2) modify or replace the softmax function to encourage non-sparse outputs.
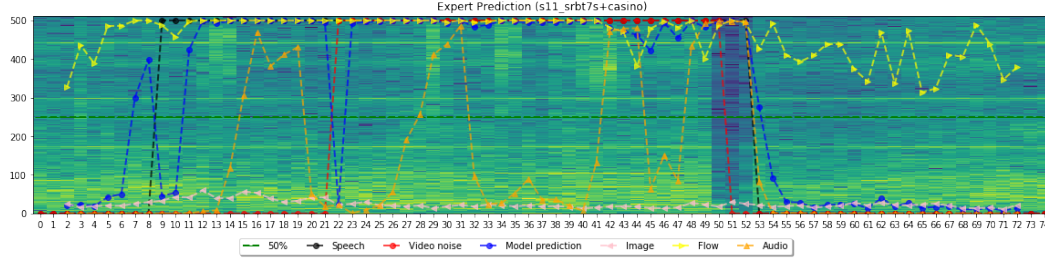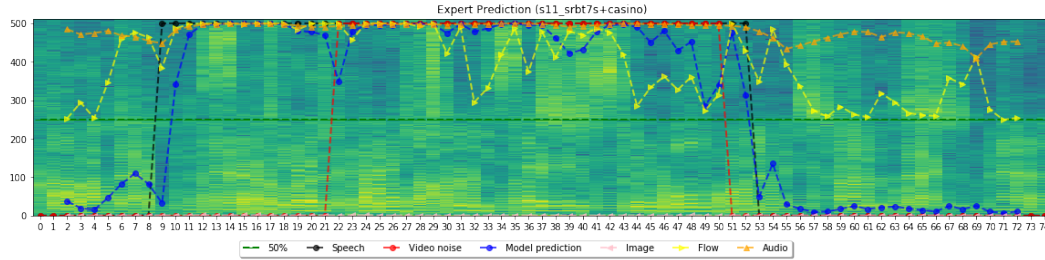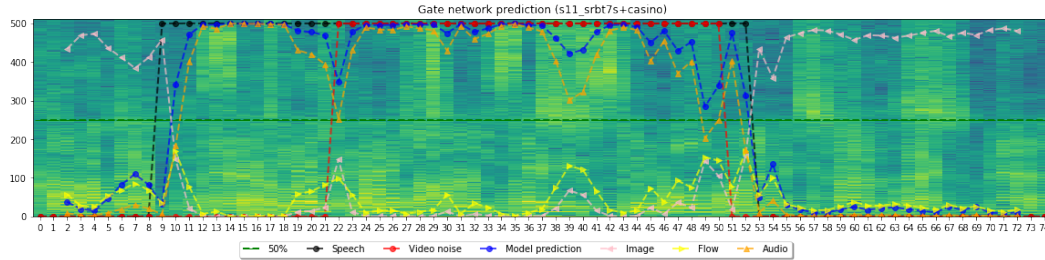
(a) Expert network outputs.
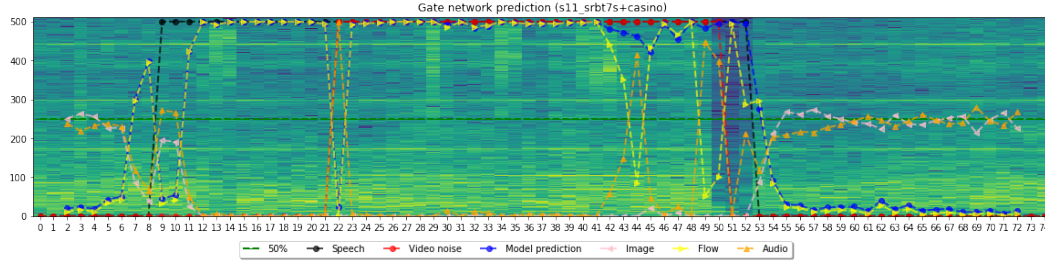
(b) Attention network outputs.

Figure 17: Model outputs (subject 11, speech srbt7s, mixed with casino noise at -5db.

(a) Expert network outputs.

(b) Attention network outputs.

Figure 18: Model outputs (subject 11, speech srbt7s, mixed with 3 random person noise at 0db.)

8 Conclusion and Future Works

In our speech activity detection experiment, we found that the best multimodal model has 95% accuracy on testing set, whereas for flow model and audio model, the accuracy is $90 - 92\%$, and 85% for image model. The multimodal model prediction is more accurate and stable than unimodal models. This suggests that multimodal model has successfully combined different sensor data for a common task. We also found the attention function dynamically estimates how reliable each expert is and assigns weights accordingly. Hence the gate module successfully opened the black-box of sensor

33

fusion, which provides insight to the relation between signal and sensor fusion process. The result shows a small step towards a structured sensor fusion method. Speech activity detection involves a sequential binary classification problem. As we discussed, another interesting problem is speech separation (e.g. denoising). We are currently working on this problem.

We found that co-learning using distance-based regularizer decreases prediction accuracy. This is possibly due to co-learning separates modality invariant features and is not designed for speech activity detection. We also proposed to combine probablistic graphical model with deep neural network to construct a model-based sensor fusion model. The model has a co-learning design which tries to separate modality-invariant and modality-dependent features. We are pursuing this direction at the moment. We would like to investigate if model-free and model-based co-learning can successfully separate modality invariant features from modality dependent features.

The combination of denoising autoendoder, siamese network, and conditional attention mixture model is another work we are currently working on as part of the author's dissertation. Although we choose audio event detection as a application, this framework is general enough to solve other problems, e.g. speaker identification and video event detection in multimodal setting. We will explore these possibilities in future.

Acknowledgement

References

Robert H Shumway and David S Stoffer. *Time series analysis and its applications: with R examples*. Springer, 2011.

Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. Multimodal machine learning: A survey and taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

Ariel Ephrat, Tavi Halperin, and Shmuel Peleg. Improved speech reconstruction from silent video. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 455–462, 2017.

Yannis M Assael, Brendan Shillingford, Shimon Whiteson, and Nando de Freitas. Lipnet: Sentence-level lipreading. *arXiv preprint arXiv:1611.01599*, 2016.

Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y Ng. Multi-modal deep learning. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 689–696, 2011.

Youssef Mroueh, Etienne Marcheret, and Vaibhava Goel. Deep multimodal learning for audio-visual speech recognition. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2130–2134. IEEE, 2015.

Weijiang Feng, Naiyang Guan, Yuan Li, Xiang Zhang, and Zhigang Luo. Audio visual speech recognition with multimodal recurrent neural networks. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 681–688. IEEE, 2017.

Jen-Cheng Hou, Syu-Siang Wang, Ying-Hui Lai, Yu Tsao, Hsiu-Wen Chang, and Hsin-Min Wang. Audio-visual speech enhancement based on multimodal deep convolutional neural network. *arXiv preprint arXiv:1709.00944*, 2017.

Ido Ariav, David Dov, and Israel Cohen. A deep architecture for audio-visual voice activity detection in the presence of transients. *Signal Processing*, 142:69–74, 2018.

DeLiang Wang and Jitong Chen. Supervised speech separation based on deep learning: An overview. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(10):1702–1726, 2018.

John R Hershey, Zhuo Chen, Jonathan Le Roux, and Shinji Watanabe. Deep clustering: Discriminative embeddings for segmentation and separation. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pages 31–35. IEEE, 2016.

Dong Yu, Morten Kolbæk, Zheng-Hua Tan, and Jesper Jensen. Permutation invariant training of deep models for speaker-independent multi-talker speech separation. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 241–245. IEEE, 2017.

Yusuf Isik, Jonathan Le Roux, Zhuo Chen, Shinji Watanabe, and John R Hershey. Single-channel multi-speaker separation using deep clustering. *arXiv preprint arXiv:1607.02173*, 2016.

Aviv Gabbay, Asaph Shamir, and Shmuel Peleg. Visual speech enhancement. *arXiv preprint arXiv:1711.08789*, 2017.

Aviv Gabbay, Ariel Ephrat, Tavi Halperin, and Shmuel Peleg. Seeing through noise: Visually driven speaker separation and enhancement. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3051–3055. IEEE, 2018.

Ariel Ephrat, Inbar Mosseri, Oran Lang, Tali Dekel, Kevin Wilson, Avinatan Hassidim, William T Freeman, and Michael Rubinstein. Looking to listen at the cocktail party: A speaker-independent audio-visual model for speech separation. *arXiv preprint arXiv:1804.03619*, 2018.

Andrew Owens and Alexei A Efros. Audio-visual scene analysis with self-supervised multisensory features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 631–648, 2018.

Ruohan Gao, Rogerio Feris, and Kristen Grauman. Learning to separate object sounds by watching unlabeled video. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 35–53, 2018.

Hang Zhao, Chuang Gan, Andrew Rouditchenko, Carl Vondrick, Josh McDermott, and Antonio Torralba. The sound of pixels. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 570–586, 2018.

Nitish Srivastava and Ruslan R Salakhutdinov. Multimodal learning with deep boltzmann machines. In *Advances in neural information processing systems*, pages 2222–2230, 2012.

Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014a.

Amirsina Torfi, Seyed Mehdi Iranmanesh, Nasser Nasrabadi, and Jeremy Dawson. 3d convolutional neural networks for cross audio-visual matching recognition. *IEEE Access*, 5:22081–22091, 2017.

Kihyuk Sohn, Wenling Shang, and Honglak Lee. Improved multimodal deep learning with variation of information. In *Advances in Neural Information Processing Systems*, pages 2141–2149, 2014.

Christopher M Bishop et al. *Pattern recognition and machine learning*, volume 1. springer New York, 2006.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014a.

Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014b.

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

Jan K Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. Attention-based models for speech recognition. In *Advances in neural information processing systems*, pages 577–585, 2015.

Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. Recurrent models of visual attention. In *Advances in neural information processing systems*, pages 2204–2212, 2014.

Jimmy Ba, Volodymyr Mnih, and Koray Kavukcuoglu. Multiple object recognition with visual attention. *arXiv preprint arXiv:1412.7755*, 2014.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pages 4960–4964. IEEE, 2016.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*, pages 2048–2057, 2015.

Yao Qin, Dongjin Song, Haifeng Chen, Wei Cheng, Guofei Jiang, and Garrison Cottrell. A dual-stage attention-based recurrent neural network for time series prediction. *arXiv preprint arXiv:1704.02971*, 2017.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

Christopher M Bishop. Mixture density networks. Technical report, Citeseer, 1994.

Marina Meila and Michael I Jordan. Learning fine motion by markov mixtures of experts. In *Advances in neural information processing systems*, pages 1003–1009, 1996.

Helen L Bear and Richard Harvey. Phoneme-to-viseme mappings: the good, the bad, and the ugly. *Speech Communication*, 95:40–67, 2017.

Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C Courville, and Yoshua Bengio. A recurrent latent variable model for sequential data. In *Advances in neural information processing systems*, pages 2980–2988, 2015.

Marco Fraccaro, Søren Kaae Sønderby, Ulrich Paquet, and Ole Winther. Sequential neural models with stochastic layers. In *Advances in neural information processing systems*, pages 2199–2207, 2016.

Lawrence R Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

Yoshua Bengio, Aaron Courville, and Pierre Vincent. Representation learning: A review and new perspectives. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(8):1798–1828, 2013.

Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

Geoffrey E Hinton and Sam T Roweis. Stochastic neighbor embedding. In *Advances in neural information processing systems*, pages 833–840, 2002.

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(2579-2605):85, 2008.

Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *Computer vision and pattern recognition, 2006 IEEE computer society conference on*, volume 2, pages 1735–1742. IEEE, 2006.

Sean Bell and Kavita Bala. Learning visual similarity for product design with convolutional neural networks. *ACM Transactions on Graphics (TOG)*, 34(4):98, 2015.

Jort F Gemmeke, Daniel PW Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R Channing Moore, Manoj Plakal, and Marvin Ritter. Audio set: An ontology and human-labeled dataset for audio events. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 776–780. IEEE, 2017.

Yichi Zhang and Zhiyao Duan. Imisound: An unsupervised system for sound query by vocal imitation. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pages 2269–2273. IEEE, 2016.

Yichi Zhang and Zhiyao Duan. Iminet: Convolutional semi-siamese networks for sound search by vocal imitation. 2017.

Shawn Hershey, Sourish Chaudhuri, Daniel PW Ellis, Jort F Gemmeke, Aren Jansen, R Channing Moore, Manoj Plakal, Devin Platt, Rif A Saurous, Bryan Seybold, et al. Cnn architectures for large-scale audio classification. In *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*, pages 131–135. IEEE, 2017.

Yusuf Aytar, Carl Vondrick, and Antonio Torralba. Soundnet: Learning sound representations from unlabeled video. In *Advances in neural information processing systems*, pages 892–900, 2016.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11(Dec):3371–3408, 2010.

Bo Li, Carolina Parada, Gabor Simko, Shuo-yiin Chang, and Tara Sainath. Endpoint detection using grid long short-term memory networks for streaming speech recognition. 2017.

Paris Smaragdis, Bhiksha Raj, and Madhusudana Shashanka. Supervised and semi-supervised separation of sounds from single-channel mixtures. *Independent Component Analysis and Signal Separation*, pages 414–421, 2007.

Lijiang Guo and Minje Kim. Bitwise source separation on hashed spectra: An efficient posterior estimation scheme using partial rank order metrics. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 761–765. IEEE, 2018.

Martin Cooke, Jon Barker, Stuart Cunningham, and Xu Shao. An audio-visual corpus for speech perception and automatic speech recognition. *The Journal of the Acoustical Society of America*, 120(5):2421–2424, 2006.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014b.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016.

1266 Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image
1267   recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*,
1268   pages 770–778, 2016.

1269 Zhiyao Duan, Gautham J Mysore, and Paris Smaragdis. Speech enhancement by online non-negative
1270   spectrogram decomposition in nonstationary noise environments. In *Thirteenth Annual Conference*
1271   *of the International Speech Communication Association*, 2012.