Neural Image and Video Understanding

by

RASOOL FAKOOR

Presented to the Faculty of the Graduate School of
The University of Texas at Arlington in Partial Fulfillment
of the Requirements
for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT ARLINGTON
August 2017

To

Amazing Azade

ACKNOWLEDGEMENTS

Foremost, I would like to express my sincere gratitude to my advisor Dr. Manfred Huber for the continuous support of my Ph.D. study and research, for his enthusiasm, motivation, and immense knowledge. I have learned a lot from him. It has been a privilege and truly an honor to have him as a mentor. Dr. Huber has been an amazing mentor and advisor. I can not thank him enough.

I would like to specifically thank Dr. Vassilis Athitsos for his support and mentorship.

I would like to thank the rest of my thesis committee: Dr. Gergely V. Zaruba and Dr. Junzhou Huang for their encouragement and insightful comments.

I would like to thank Dr. Mohit Bansal and Dr. Matthew R. Walter for their guidance and help during my time at TTIC and wonderful mentorship and collaboration.

I would like to thank Dr. Pushmeet Kohli, Dr. Abdel-rahman Mohamed, and Dr. Margaret Mitchell for their guidance and help during and after my time at MSR and wonderful mentorship and collaboration. I learn a lot from them.

I thank my friends Saravanan and Farzin for all fun time that we had together.

Special thank to Azade without her love, support, and encouragement, I would not have finished this journey.

August 18, 2017

ABSTRACT

Neural Image and Video Understanding

RASOOL FAKOOR, Ph.D.

The University of Texas at Arlington, 2017

Supervising Professor: Manfred Huber

Even though recent works on neural architectures have shown promising results at tasks like image recognition, object detection, playing Atari games, etc., learning a mapping from a visual space to a language space or vice versa remains challenging in problems like image/video captioning or question-answering tasks. Furthermore, transferring knowledge between seen and unseen classes in a setting like zero-shot learning is quite challenging given the fact that a model should be able to make a prediction for novel test data belonging to classes for which no examples have been seen during training.

To address these issues, this dissertation will first introduce a novel memory-based attention model for video description. Specifically, attention-based models have shown promising and interesting results for image captioning. However, they are not able to model the higher-order interactions involved in problems such as video description/captioning, where the relationship between parts of the video and the concepts being depicted is complex. The proposed model here utilizes memories of past attention when reasoning about where to attend to, in the current time step.

Secondly, this dissertation will introduce an end-to-end deep neural network model for attribute-based zero-shot learning with layer-specific regularization that encourages the higher, class-level layers to generalize beyond the training classes. This architecture enables the model to 'transfer' knowledge learned from seen training images to a set of novel, unseen test images.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

Figure Page

LIST OF TABLES

CHAPTER 1

Introduction

Machine learning in image and video domains has made significant strides in recent years, leading to many systems that allow to 'interpret' images or video either by classifying their content, by detecting object inside them, or by generating language representations of their content. However, many of these systems still require significant problem-specific expert engineering of the structure to be successful and often do not make the most efficient use of the available data. The work in this dissertation focuses on addressing two main issues:

- The efficient incorporation of iterative attention and differentiable memory structures into deep network architectures.
- The automatic imposition of known correlation structures into deep networks

Both of these mechanisms are aimed at allowing deep models to more fully take advantage of available data in end-to-end learning systems.

1.1 Iterative attention and differentiable memory

In recent years, we have witnessed remarkable progress in machine learning based methods in which they try to learn a mapping from visual space (i.e., video or image) to a language space (i.e. natural language). Image and video captioning is one such application. One of the primary challenges in learning a mapping from a visual space to a language space is learning a representation that not only effectively represents each of these modalities, but is also able to translate a representation from one space to the other. One of the emerging paradigms, shared by models for these kinds of problems (e.g image/video captioning), is the notion of an attention mechanism that guides the model to attend to certain parts of the image while generating. The attention models used for problems such as image captioning typically depend on the single image under consideration and the partial output generated so far, jointly capturing one region of an image and the words being generated. However, such models cannot directly capture the temporal reasoning necessary to effectively produce words that refer to actions and events taking place over multiple frames in a video.

Motivated by these observations, we present a method to improve video description generation by modeling higher-order interactions between video frames and described concepts (Fakoor et al., 2016a). By storing past visual attention in the video associated to previously generated words, the system is able to decide what to look at and describe in light of what it has already looked at and described. This enables not only more effective local attention, but tractable consideration of the video sequence while generating each word. Evaluation on the challenging and popular *MSVD* and *Charades* datasets demonstrates that the proposed architecture outperforms previous video description approaches without requiring external temporal video features.

In Chapter 2, we introduce this novel memory-based attention model for video description (Fakoor et al., 2016a). In addition, in Chapter 2, we study related works and conduct extensive experiments to show the effectiveness of our proposed approach.

1.2 Imposing structure into deep networks

The availability of large image datasets has been integral to recent progress on image classification. However, many approaches to image classification make the assumption that the classes encountered at test time are a subset of those seen during training. Zero-shot learning considers the problem of classifying images of novel, unseen classes by transferring knowledge learned from a separate space of training classes. An effective strategy is to learn the relationship between image features and classes via intermediate semantic representations such as attribute signatures (e.g., color and shape). We propose a deep neural network model for attribute-based zero-shot learning with layer-specific regularization that encourages the higher, class-level layers to generalize beyond the training classes (Fakoor et al., 2016b, 2015). This architecture enables our model to transfer knowledge learned from seen training images to a set of novel, unseen test images. We evaluate our method on a number of benchmark datasets and achieve results that equal or exceed state-of-the-art techniques. We then conduct a series of ablations to elucidate the contributions of layer-specific regularization and the architecture depth. Results on several benchmark datasets demonstrate that our method achieves greater transferability than existing state-of-the-art methods.

In Chapter 3, we discuss this proposed model in details and we provide rigorous analysis and extensive experiments over the popular dataset and show the effectiveness of our models (Fakoor et al., 2016b, 2015). In addition, we show the scenarios in which our proposed model fail.

CHAPTER 2

Memory-augmented Attention Modelling for Videos (Fakoor et al., 2016a)

Deep neural architectures have led to remarkable progress in computer vision and natural language processing problems. Image captioning is one such problem, where the combination of convolutional structures (Krizhevsky et al., 2012; LeCun et al., 1998), and sequential recurrent structures (Sutskever et al., 2014) leads to remarkable improvements over previous work Fang et al. (2015); Devlin et al. (2015). One of the emerging modelling paradigms, shared by models for image captioning as well as related vision-language problems, is the notion of an attention mechanism that guides the model to attend to certain parts of the image while generating Xu et al. (2015a).

The attention models used for problems such as image captioning typically depend on the single image under consideration and the partial output generated so far, jointly capturing one region of an image and the words being generated. However, such models cannot directly capture the temporal reasoning necessary to effectively produce words that refer to actions and events taking place over multiple frames in a video. For example, in a video depicting “someone waving a hand”, the “waving” action can start from any frame and can continue on for a variable number of following frames. At the same time, videos contain many frames that do not provide additional information over the smaller set of frames necessary to generate a summarizing description. Given these challenges, it is not surprising that even with recent advancements in image captioning Fang et al. (2015); Xu et al. (2015a); Johnson et al. (2016); Vinyals et al. (2015); Donahue et al. (2015), video captioning has remained challenging.

Motivated by these observations, we introduce a memory-based attention mechanism for video captioning and description. Our model utilizes memories of past attention in the video when reasoning about where to attend in a current time step. This allows the model to not only effectively leverage local attention, but also to consider the entire video as it generates each word. This mechanism effectively binds information from both vision and language sources into a coherent structure.

Our work shares the same goals as recent work on attention mechanisms for sequence-to-sequence architectures, such as Rocktäschel et al. (2016) and Yang et al. (2016). Rocktäschel et al. (2016) consider the domain of entailment relations, where the goal is to determine entailment given two input sentences. They propose a soft attention model that is not only focused on the current state, but the previous as well. In our model, all previous attentions are explicitly stored into memory, and the system learns to memorize the encoded version of the input videos conditioned on previously seen words. Yang et al. (2016) and our work both try to solve the problem of locality of attention in vision-to-language, but while Yang et al. (2016) introduce a memory architecture optimized for single image caption generation, we introduce a memory architecture that operates on a streaming video's temporal sequence.

The contributions of this work include:

- A deep learning architecture that represents video with an explicit model of the video's temporal structure.
- A method to jointly model the video description and temporal video sequence, connecting the visual video space and the language description space.
- A memory-based attention mechanism that learns iterative attention relationships in a simple and effective sequence-to-sequence memory structure.

- Extensive comparison of this work and previous work on the video captioning problem on the MSVD (Chen and Dolan, 2011) and the Charades (Sigurdsson et al., 2016) datasets.

We focus on the video captioning problem, however, the proposed model is general enough to be applicable in other sequence problems where attention models are used (e.g., machine translation or recognizing entailment relations).

2.1 Related Work

One of the primary challenges in learning a mapping from a visual space (i.e., video or image) to a language space is learning a representation that not only effectively represents each of these modalities, but is also able to translate a representation from one space to the other. Rohrbach et al. (2013a) developed a model that generates a semantic representation of visual content that can be used as the source language for the language generation module. Venugopalan et al. (2015b) proposed a deep method to translate a video into a sentence where an entire video is represented with a single vector based on the mean pool of frame features. However, it was recognized that representing a video by an average of its frames loses the temporal structure of the video. To address this problem, recent work (Yao et al., 2015; Pan et al., 2016a; Venugopalan et al., 2015a; Andrew Shin, 2016; Pan et al., 2016b; Xu et al., 2015b; Ballas et al., 2016; Yu et al., 2016) proposed methods to model the temporal structure of videos as well as language.

The majority of these methods are inspired by sequence-to-sequence (Sutskever et al., 2014) and attention (Bahdanau et al., 2015) models. Sequence learning was proposed to map the input sequence of a source language to a target language (Sutskever et al., 2014). Applying this method with an additional attention mechanism to the problem of translating a video to a description showed promising initial results, however, revealed additional challenges. First, modelling the video content with a fixed-length vector in order to map it to a language space is a more complex problem than mapping from a language to a language, given the complexity of visual content and the difference between the two modalities. Since not all frames in a video are equally salient for a short description, and an event can happen in multiple frames, it is important for a model to identify which frames are most salient. Further, the models need additional work to be able to focus on points of interest within the video frames to select what to talk about. Even a variable-length vector to represent a video using attention (Yao et al., 2015) can have some problems.

More specifically, current attention methods are local Yang et al. (2016), since the attention mechanism works in a sequential structure, and lack the ability to capture global structure. Moreover, combining a video and a description as a sequence-to-sequence problem motivates using some variant of a recurrent neural network (RNN) (Hochreiter and Schmidhuber, 1997): Given the limited capacity of a recurrent network to model very long sequences, memory networks (Weston et al., 2014; Sukhbaatar et al., 2015) have been introduced to help the RNN memorize sequences. However, one problem these memory networks suffer from is difficulty in training the model. The model proposed by Weston et al. (2014) requires supervision at each layer, which makes training with backpropagation a challenging task. Sukhbaatar et al. (2015) proposed a memory network that can be trained end-to-end, and the current work follows this research line to tackle the challenging problem of modeling vision and language memories for video description generation.

2.2 Learning to Attend and Memorize

A main challenge in video description is to find a mapping that can capture the connection between the video frames and the video description. Sequence-to-sequence models, which work well at connecting input and output sequences in machine transla-

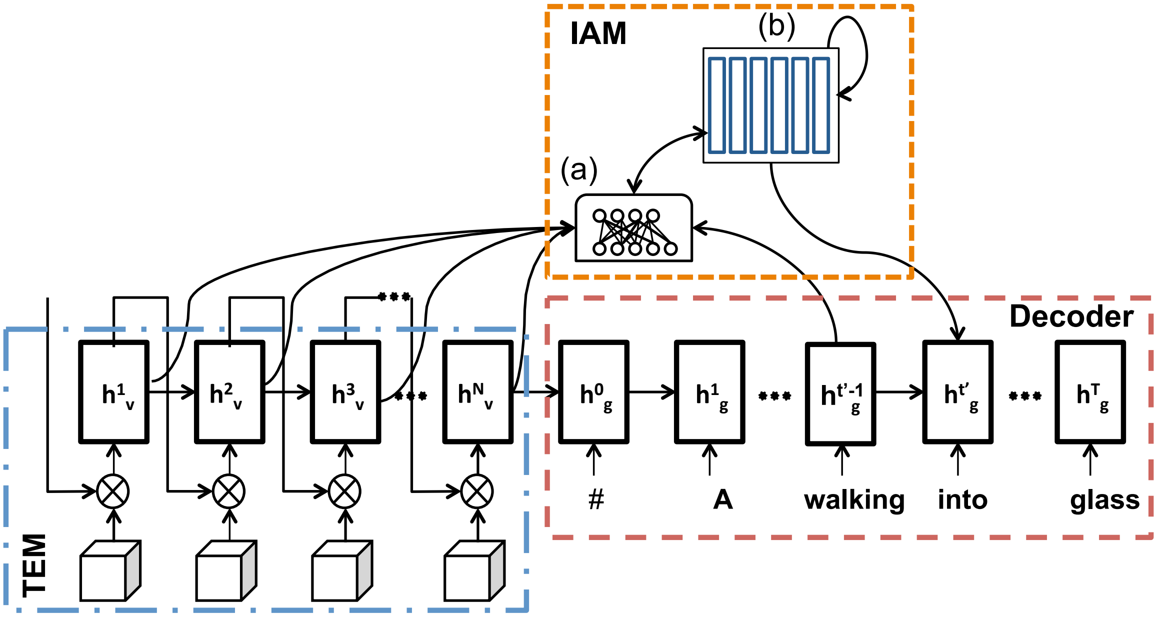


Figure 2.1. Our proposed architecture. Each component of our model is described in 2.2.1 through 2.2.3..

tion (Sutskever et al., 2014), do not perform as well for this task, as there is not the same direct alignment between a full video sequence and its summarizing description.

Our goal in the video description problem is to create an architecture that learns which moments to focus on in a video sequence in order to generate a summarizing natural language description. The modelling challenges we set forth for the video description problem are: (1) Processing the temporal structure of the video; (2) Learning to attend to important parts of the video; and (3) Generating a description where each word is relevant to the video. At a high-level, this can be understood as having three primary parts: *When* moments in the video are particularly salient; *what* concepts to focus on; and *how* to talk about them. We directly address these issues in an end-to-end network with three primary corresponding components (Figure 2.1): A Temporal Model (TEM), An Iterative Attention/Memory Model (IAM), and a Decoder. In summary:

- **When:** Frames within the video sequence - The Temporal Model (TEM).

- **What:** Language-grounded concepts depicted in the video - The Iterative Attention/Memory mechanism (IAM).
- **How:** Words that fluently describe the *what* and *when* - The Decoder.

The Temporal Model is in place to capture the temporal structure of the video: It functions as a *when* component. The Iterative Attention/Memory is a main contribution of this work, functioning as a *what* component to remember relationships between words and video frames, and storing longer term memories. The Decoder generates language, and functions as the *how* component to create the final description.

To train the system end to end, we formulate the problem as sequence learning to maximize the probability of generating a correct description given a video:

$$\Theta^* = \arg\max_{\Theta} \sum_{(S, f_1, \ldots, f_N)} \log \; p(S | f_1, \ldots, f_N; \mathbf{\Theta}) \tag{2.1}$$

where S is the description, $f_1, f_2, \ldots, f_N$ are the input video frames, and Θ is the model parameter vector. In the next sections, we will describe each component of the model, then explain the details of training and inference.

2.2.0.0.1 Notational note: Numbered equations use bold face to denote multi-dimensional learnable parameters, e.g., $\mathbf{W_p^j}$. To distinguish the two different sets of time steps, one for video frames and one for words in the description, we use the notation t for video and t' for language. Throughout, the terms *description* and *caption* are used interchangeably.

2.2.1 Temporal Model (TEM)

The first module we introduce encodes the temporal structure of the input video. A clear framework to use for this is a Recurrent Neural Network (RNN), which has been shown to be effectual in modelling the temporal structure of sequential data such as video (Ballas et al., 2016; Sharma et al., 2015; Venugopalan et al., 2015a) and speech (Graves and Jaitly, 2014). In order to apply this in video sequences to generate a description, we seek to capture the fact that frame-to-frame temporal variation tends to be local (Brox and Malik, 2011) and critical in modeling motion (Ballas et al., 2016). Visual features extracted from the last fully connected layers of Convolutional Neural Networks (CNNs) have been shown to produce state-of-the-art results in image classification and recognition (Simonyan and Zisserman, 2014; He et al., 2016), and thus seem a good choice for modeling visual frames. However, these features tend to discard low level information useful in modeling the motion in the video (Ballas et al., 2016).

To address these challenges, we implement an RNN we call the Temporal Model (TEM). At each time step of the TEM, a video frame encoding from a CNN serves as input. Rather than extracting video frame features from a fully connected layer of the pretrained CNN, we extract intermediate convolutional maps.

In detail, for a given video X with N frames $X = [X^1, X^2, \cdots, X^N]$, N convolutional maps of size $R^{L \times D}$ are extracted, where L is the number of locations in the input frame and D is the number of dimensions (See TEM in Figure 2.1). To enable the network to store the most important L locations of each frame, we use a soft location attention mechanism, $f_{\mathbf{Latt}}$ (Bahdanau et al., 2015; Xu et al., 2015a; Sharma et al., 2015). We first use a softmax to compute L probabilities that specify the importance of different parts in the frame, and this creates an input map for $f_{\mathbf{Latt}}$.

Formally, given a video frame at time t, $X^t \in R^{L \times D}$, the $f_{\mathbf{Latt}}$ mechanism is defined as follows:

$$\rho_j^t = \frac{\exp(\mathbf{W_p^j} h_v^{t-1})}{\sum_{k=1}^{L} \exp(\mathbf{W_p^k} h_v^{t-1})} \tag{2.2}$$

$$f_{\mathbf{Latt}}(X^t, h_v^{t-1}; \mathbf{W_p}) = \sum_{j=1}^{L} \rho_j^t X_j^t \tag{2.3}$$

where $h_v^{t-1} \in R^K$ is the hidden state of the TEM at time t-1 with K dimensions, and $W_p \in R^{L \times K}$. For each video frame time step, TEM learns a vector representation by applying location attention on the frame convolution map, conditioned on all previously seen frames:

$$F^t = f_{\mathbf{Latt}}(X^t, h_v^{t-1}; \mathbf{W_p}) \tag{2.4}$$

$$h_v^t = f_{\mathbf{v}}(F^t,\ h_v^{t-1}; \mathbf{\Theta_v}) \tag{2.5}$$

where $f_{\mathbf{v}}$ can be an RNN/LSTM/GRU cell and Θ_v is the parameters of the $f_{\mathbf{v}}$. Due to the fact that vanilla RNNs have gradient vanishing and exploding problems (Pascanu et al., 2013), we use gradient clipping, and an LSTM with the following flow to handle potential vanishing gradients:

$$\begin{aligned}
i^t &= \sigma(F^t\mathbf{W_{x_i}} + (h_v^{t-1})^T\mathbf{W_{h_i}}) \\
f^t &= \sigma(F^t\mathbf{W_{x_f}} + (h_v^{t-1})^T\mathbf{W_{h_f}}) \\
o^t &= \sigma(F^t\mathbf{W_{x_o}} + (h_v^{t-1})^T\mathbf{W_{h_o}}) \\
g^t &= \tanh(F^t\mathbf{W_{x_g}} + (h_v^{t-1})^T\mathbf{W_{h_g}}) \\
c_v^t &= f^t \odot c_v^{t-1} + i^t \odot g^t \\
h_v^t &= o_t \odot \tanh(c^t)
\end{aligned}$$

where $W_{h*} \in R^{K \times K}$, $W_{x*} \in R^{D \times K}$, and we define $\Theta_v = \{W_{h*}, W_{x*}\}$.

2.2.2 Iterative Attention/Memory (IAM)

A main contribution of this work is a global view for the video description task: A memory-based attention mechanism that learns iterative attention relationships in an efficient sequence-to-sequence memory structure. We refer to this as the Iterative Attention/Memory mechanism (IAM), and it aggregates information from previously generated words and all input frames.

The IAM component is an iterative memorized attention between an input video and a description. More specifically, it learns a iterative attention structure for where to attend in a video given all previously generated words (from the Decoder), and previous states (from the TEM). This functions as a memory structure, remembering encoded versions of the video with corresponding language, and in turn, enabling the Decoder to access the full encoded video and previously generated words as it generates new words.

This component addresses several key issues in generating a coherent video description. In video description, a single word or phrase often describes action spanning multiple frames within the input video. By employing the IAM, the model can effectively capture the relationship between a relatively short bit of language and an action that occurs over multiple frames. This also functions to directly address the problem of identifying which parts of the video are most relevant for description.

The proposed Iterative Attention/Memory mechanism is formalized with an **Attention** update and a **Memory** update, detailed in Figure 2.2. Figure 2.1 illustrates where the IAM sits within the full model, with the Attention module shown in 2.1a and the Memory module shown in 2.1b.

As formalized in Figure 2.2, the *Attention* update $\hat{F}(\Theta_a)$ computes the set of probabilities in a given time step for attention within the input video states, the memory state, and the decoder state. The *Memory* update stores what has been attended to and described. This serves as the memorization component, combining the previous memory with the current iterative attention $\hat{F}$. We use an LSTM f_m with the equations described above to enable the network to learn multi-layer attention over the input video and its corresponding language. The output of this function is then used as input to the Decoder.

- Given:

  $N =$ Number of frames in a given video

  $T =$ Number of words in description

  $H_v =$ Input video states, $[h_v^1, ..., h_v^N]$

  $H_g^{t'-1} =$ Decoder state h_g at time t-1, repeated N times

  $H_m^{t'-1} =$ Memory state h_m at time t-1, repeated N times

  $W_v, W_g \in R^{K \times K}$

  $W_m \in R^{M \times K}$

  $u \in R^K$

  α = Probability over all N frames

  $\Theta_a = \{W_v, W_g, W_m, u\}$

- Attention update $[\hat{F}(\mathbf{\Theta_a})]$:

$$Q_A = \tanh(H_v \mathbf{W_v} + H_g^{t'-1} \mathbf{W_g} + H_m^{t'-1} \mathbf{W_m}) \tag{2.6}$$

$$\alpha_{t'} = \mathrm{softmax}(Q_A \mathbf{u}) \tag{2.7}$$

$$\hat{F} = H_v^T \alpha_{t'} \tag{2.8}$$

- Memory update:

$$h_m^{t'} = f_m(h_m^{t'-1},\ \hat{F}; \mathbf{\Theta_m}) \tag{2.9}$$

Figure 2.2. Iterative Attention and Memory (IAM) is formulated as an Attention update and a Memory update..

2.2.3 Decoder

In order to generate a new word conditioned on all previous words and IAM states, a recurrent structure is modelled as follows:

$$h_g^{t'} = f_g(s^{t'},\ h_m^{t'},\ h_g^{t'-1}; \mathbf{\Theta_g}) \tag{2.10}$$

$$\hat{s}^{t'} = \mathrm{softmax}((h_g^{t'})^T \mathbf{W_e}) \tag{2.11}$$

where $h_g^{t'} \in R^K$, $s^{t'}$ is a word vector at time t', $W_e \in R^{K \times |V|}$, and $|V|$ is the vocabulary size. In addition, $\hat{s}_{t'}$ assigns a probability to each word in the language. f_g is an LSTM where $s^{t'}$ and $h_m^{t'}$ are inputs and $h_g^{t'}$ is the recurrent state.

2.2.4 Training and Optimization

The goal in our network is to predict the next word given all previously seen words and an input video. In order to optimize our network parameters $\Theta = \{W_p, \Theta_v, \Theta_a, \Theta_m, \Theta_g, W_e\}$, we minimize a negative log likelihood loss function:

$$L(S, X; \mathbf{\Theta}) = -\sum_{t'}^{T} \sum_{i}^{|V|} s_i^{t'} \log(\hat{s}_i^{t'}) + \lambda \parallel \Theta \parallel_2^2 \tag{2.12}$$

where $|V|$ is the vocabulary size. We fully train our network in an *end-to-end* fashion using first-order stochastic gradient-based optimization method with an adaptive learning rate. More specifically, in order to optimize our network parameters, we use Adam (Kingma and Ba, 2015a) with learning rate 2×10^{-5} and set β_1, β_2 to 0.8 and 0.999, respectively. During training, we use a batch size of 16. The source code for this paper is available on `https://github.com/rasoolfa/videocap`.

2.3 Experiments

Dataset We evaluate the model on the *Charades* (Sigurdsson et al., 2016) dataset and the *Microsoft Video Description Corpus (MSVD)* (Chen and Dolan, 2011). Charades contains $9,848$ videos (in total) and provides $27,847$[1] video descriptions. We follow the same train/test splits as Sigurdsson et al. (2016), with 7569 train, $1,863$ test, and 400 validation. A main difference between this dataset and others is that it uses a “Hollywood in Homes”

[1] Only 16087 out of $27,847$ are used as descriptions for our evaluation since the $27,847$ refers to script of the video as well as descriptions.

approach to data collection, where "actors" are crowdsourced to act out different actions. This yields a diverse set of videos, with each containing a specific action.

MSVD is a set of YouTube videos annotated by workers on Mechanical Turk,[2] who were asked to pick a video clips representing an activity. In this dataset, each clip is annotated by multiple workers with a single sentence. The dataset contains $1,970$ videos and about $80,000$ descriptions, where $1,200$ of the videos are training data, 670 test, and the rest (100 videos) for validation. In order for the results to be comparable to other approaches, we follow the ***exact*** training/validation/test splits provided by Venugopalan et al. (2015b).

Evaluation metrics We report results on the video description generation task. In order to evaluate descriptions generated by our model, we use model-free automatic evaluation metrics. We adopt METEOR, BLEU-N, and CIDEr metrics available from the Microsoft COCO Caption Evaluation code[3] to score the system.

Video and Caption preprocessing We preprocess the captions for both datasets using the Natural Language Toolkit (NLTK)[4] and clip each description up to 30 words, since the majority have less. We extract sample frames from each video and pass each frame through VGGnet (Simonyan and Zisserman, 2014) without fine-tuning. For the experiments in this paper, we use the feature maps from *conv5_3* layer after applying *ReLU*. The feature map in this layer is $14 \times 14 \times 512$. Our TEM component operates on the flattened 196×512 of this feature cubes. For the ablation studies, features from the fully connected layer with 4096 dimensions are used as well.

[2] https://www.mturk.com/mturk/welcome

[3] https://github.com/tylin/coco-caption

[4] http://www.nltk.org/

Hyper-parameter optimization We use random search (Bergstra and Bengio, 2012) on the validation set to select hyper-parameters on both datasets. The word-embedding size, hidden layer size (for both the TEM and the Decoder), and memory size of the best model on Charades are: 237, 1316, and 437, respectively. These values are 402, 1479, and 797 for the model on the MSVD dataset. A stack of two LSTMs are used in the Decoder and TEM. The number of frame samples is a hyperparameter which is selected among 4, 8, 16, 40 on the validation set. ATT + NO TEM and NO IAM + TEM get the best results on the validation set with 40 frames, and we use this as the number of frames for all models in the ablation study.

2.3.1 Video Caption Generation

We first present an ablation analysis to elucidate the contribution of the different components of our proposed model. Then, we compare the overall performance of our model to other recent models.

Ablation Analysis

Ablation results are shown in Table 2.1, evaluating on the MSVD test set. The first (ATT + NO TEM) corresponds to a simpler version of our model in which we remove the TEM component and instead pass each frame of the video through a CNN, extracting features from the last fully-connected hidden layer. In addition, we replace our IAM with a simpler version where the model only memorizes the current step instead of all previous steps. In the next variation (ATT + TEM), it is same as the first one except we use TEM instead of fully connected CNN features. In the next ablation (NO IAM + TEM), we remove the IAM component from our model and keep the rest of the model as-is. In the next variation (IAM + NO TEM), we remove the TEM and calculate features for each frame,

Method	METEOR	BLEU-1	BLEU-2	BLEU-3	BLEU-4	CIDEr
ATT + NO TEM	31.20	77.90	65.10	55.30	44.90	**63.90**
ATT + TEM	31.00	79.00	66.50	56.30	45.50	61.00
NO IAM + TEM	30.50	78.10	65.20	55.10	44.60	60.50
IAM + NO TEM	31.00	78.70	66.90	**57.40**	**47.00**	62.10
IAM + TEM [40F]	31.70	79.00	66.20	56.0	45.60	62.20
IAM + TEM [8F]	**31.80**	**79.40**	**67.10**	56.80	46.10	62.70

Table 2.1. Ablation of proposed model with and without the IAM component on the MSVD test set.

similar to ATT + NO TEM. Finally, the last row in the table is our proposed model (IAM + TEM) with all its components.

The IAM plays a significant role in the proposed model, and removing it causes a large drop in performance, as measured by both BLEU and METEOR. On the other hand, removing the TEM by itself does not drop performance as much as dropping the IAM. Putting the two together, they complement one another to result in overall better performance for METEOR. However, further development on the TEM component in future work is warranted. In the NO IAM + TEM condition, an entire video must be represented with a fixed-length vector, which may contribute to the lower performance (Bahdanau et al., 2015). This is in contrast to the other models, which apply single layer attention or IAM to search relevant parts of the video aligned with the description.

Performance Comparison

To extensively evaluate the proposed model, we compare with state-of-the-art models and baselines for the video caption generation task on the MSVD dataset. In this experiment, we use 8 frames per video as the inputs to the TEM module. As shown in Table

Method	METEOR	BLEU-1	BLEU-2	BLEU-3	BLEU-4	CIDEr
Venugopalan et al. (2015b)	27.7	–	–	–	–	–
Venugopalan et al. (2015a)	29.2	–	–	–	–	–
Pan et al. (2016b)	29.5	74.9	60.9	50.6	40.2	–
Yu et al. (2016)	31.10	77.30	64.50	54.60	44.30	–
Pan et al. (2016a)	**33.10**	79.20	66.30	55.10	43.80	–
Our Model	31.80	**79.40**	**67.10**	**56.80**	**46.10**	**62.70**
Yao et al. (2015) + C3D	29.60	–	–	–	41.92	51.67
Venugopalan et al. (2015a) + Flow	29.8	–	–	–	–	–
Ballas et al. (2016) + FT	30.75	–	–	–	49.0	59.37
Pan et al. (2016b) + C3D	31.0	78.80	66.0	55.4	45.3	–
Yu et al. (2016) + C3D	32.60	81.50	70.40	60.4	49.90	–

Table 2.2. Video captioning evaluation on MSVD (670 videos).

Method	METEOR	BLEU-1	BLEU-2	BLEU-3	BLEU-4	CIDEr
Human (Sigurdsson et al., 2016)	24	62	43	29	20	53
Sigurdsson et al. (2016)	16	49	30	18	11	14
Our Model	**17.6**	**50**	**31.1**	**18.8**	**11.5**	**16.7**

Table 2.3. Video captioning evaluation on Charades (1863 videos). Sigurdsson et al. (2016) results use the Venugopalan et al. (2015a) model.

2.2,[5] our proposed model achieves state-of-the-art scores in BLEU-4, and outperforms almost all systems on METEOR. The closest-scoring comparison system, from Pan et al. (2016a), shows a trade-off between METEOR and BLEU: BLEU prefers descriptions with short-distance fluency and high lexical overlap with the observed descriptions, while METEOR permits less direct overlap and longer descriptions. A detailed study of the generated descriptions between the two systems would be needed to better understand these differences.

[5] The symbol – indicates that the score was not reported by the corresponding paper. The horizontal line in Table 2.2 separates models that do/do not use external features for the video representation.

The improvement over previous work is particularly noteworthy because we do not use external features for the video, such as Optical Flow (Brox et al., 2004) (denoted Flow), 3-Dimensional Convolutional Network features (Tran et al., 2015) (denoted C3D), or fine-tuned CNN features (denoted FT), which further enhances aspects such as action recognition by leveraging an external dataset such as UCF-101. The only system using external features that outperforms the model proposed here is from Yu et al. (2016), who uses a slightly different version of the same dataset[6] along with C3D features for a large improvement in results (compare Table 2.2 rows 4 and 11); future work may explore the utility of external visual features for this work. Here, we demonstrate that the proposed architecture maps visual space to language space with improved performance over previous work, before addition of further resources.

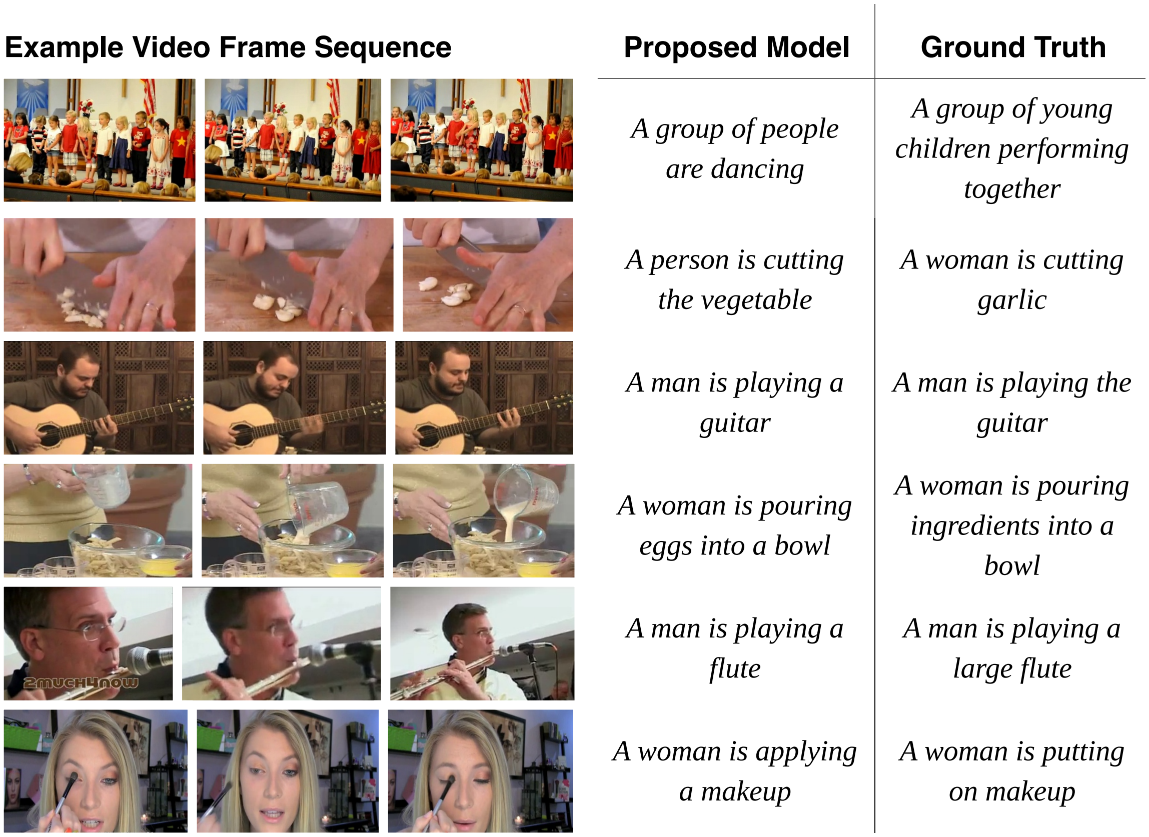
We additionally report results on the Charades dataset Sigurdsson et al. (2016), which is challenging to train on because there are only a few (≈ 2) captions per video. In this experiment, we use 16 frames per video as the input to the TEM module. As shown in Table 2.3, our method achieves a 10% relative improvement over the Venugopalan et al. (2015a) model reported by Sigurdsson et al. (2016). It is worth noting that humans reach a METEOR score of 24 and a BLEU-4 score of 20, illustrating the low upper bound in this task.[7]

Results Discussion

We show some example descriptions generated by our system in Figures 2.3 and 2.4. The model generates mostly correct descriptions, with naturalistic variation from the ground truth. Errors illustrate a preference to describe items that have a higher likelihood

[6] Yu et al. (2016) uses the MSVD dataset reported in Guadarrama et al. (2013), which has different preprocessing.

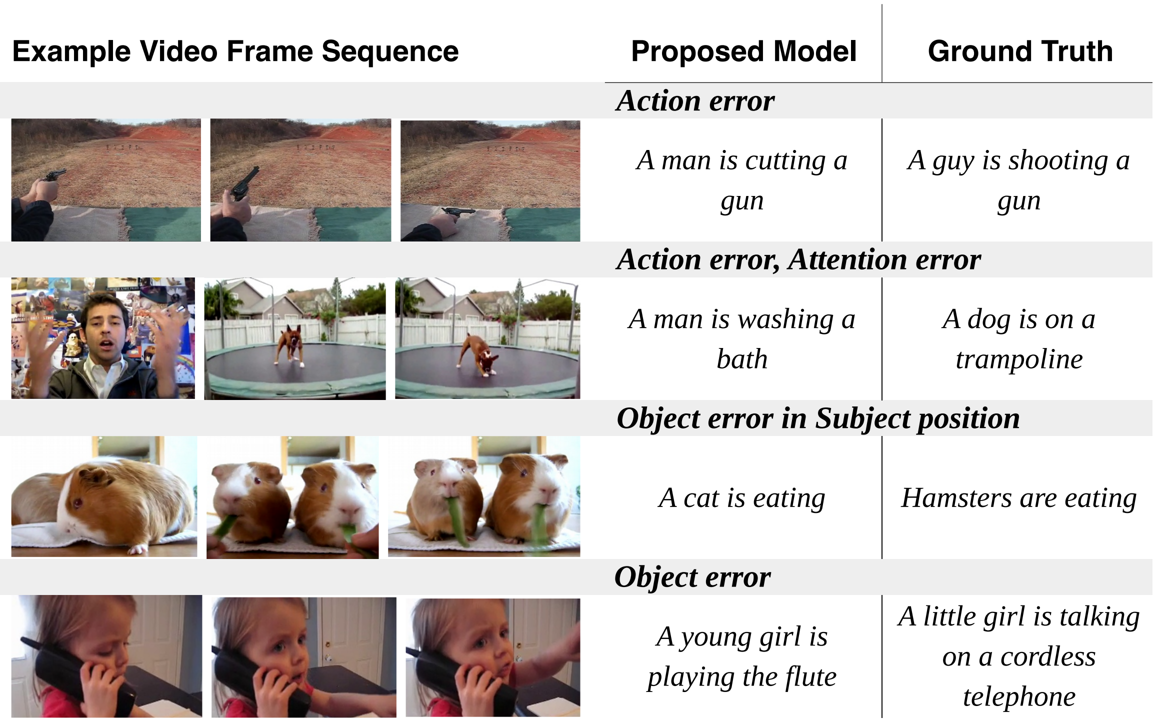
[7] For comparison, the upper bound BLEU score in machine translation for English to French is above 30.



Example Video Frame Sequence	Proposed Model	Ground Truth
	A group of people are dancing	*A group of young children performing together*
	A person is cutting the vegetable	*A woman is cutting garlic*
	A man is playing a guitar	*A man is playing the guitar*
	A woman is pouring eggs into a bowl	*A woman is pouring ingredients into a bowl*
	A man is playing a flute	*A man is playing a large flute*
	A woman is applying a makeup	*A woman is putting on makeup*

Figure 2.3. Example captions generated successfully by our model on MSVD test videos.

of being mentioned, even if they appear in less of the frames. For example, in the "a dog is on a trampoline" video, our model focuses on the man, who appears in only a few frames, and generates the incorrect description "a man is washing a bath". The errors, alongside the ablation study shown in Table 2.1, suggest that the TEM module in particular may be further improved by focusing on how frames in the video sequence are captured and passed to the IAM module.



Example Video Frame Sequence	Proposed Model	Ground Truth
Action error		
	A man is cutting a gun	*A guy is shooting a gun*
Action error, Attention error		
	A man is washing a bath	*A dog is on a trampoline*
Object error in Subject position		
	A cat is eating	*Hamsters are eating*
Object error		
	A young girl is playing the flute	*A little girl is talking on a cordless telephone*

Figure 2.4. Example captions in which our model made mistakes on MSVD test videos.

2.4 Conclusion

We introduce a general framework for an memory-based sequence learning model, trained end-to-end. We apply this framework to the task of describing an input video with a natural language description. Our model utilizes a deep learning architecture that represents video with an explicit model of the video's temporal structure, and jointly models the video description and the temporal video sequence. This effectively connects the visual video space and the language description space.

A memory-based attention mechanism helps guide where to attend and what to reason about as the description is generated. This allows the model to not only reason efficiently about local attention, but also to consider the full sequence of video frames during the generation of each word. Our experiments confirm that the memory components in our architecture, most notably from the IAM module, play a significant role in improving the performance of the entire network.

Future work should raim to refine the temporal video frame model, TEM, and explore how to improve performance on capturing the ideal frames for each description.

CHAPTER 3

Deep Transferable Zero-shot Learning with Attribute-based Regularization and Embeddings (Fakoor et al., 2016b, 2015)

The availability of large image datasets (Deng et al., 2009) has been integral to recent progress on image classification within the computer vision community (Krizhevsky et al., 2012). However, many approaches to image classification make the assumption that the classes encountered at test time are a subset of those seen during training. Zero-shot learning (ZSL) removes this assumption by considering test images with novel classes for which no examples have been seen during training. This problem can be addressed by transferring knowledge gleaned from the "seen" training image classes to the "unseen" test image classes based on learned, generalizable relationships between an image and its corresponding class. One way of modeling this interaction is through the use of mid-level semantic representations, such as human-labeled attributes (e.g., color and shape) (Lampert et al., 2009), that can then be inferred for test classes and, in turn, used for classification.

A number of methods exist that leverage the use of attributes as a form of transferable knowledge for zero-shot learning (Lampert et al., 2009, 2014; Romera-paredes and Torr, 2015; Huang et al., 2015; Zhang and Saligrama, 2015; Akata et al., 2015). One class of techniques assumes that the attributes are independent given the class (Lampert et al., 2014, 2009). After learning a model for attribute prediction, they then use these predictions in an attribute-based classifier to predict novel image classes. However, ignoring the dependence among attributes results in weaker performance and, in turn, the loss of transferability when compared to models that explicitly account for these dependencies (Huang et al., 2015; Zhang and Saligrama, 2015; Akata et al., 2015; Romera-paredes and Torr, 2015).

The challenge here is to encode as much knowledge as possible from the seen classes, while keeping the semantic representations sufficiently generic in order to predict unseen classes at test time. However, designing a model that is able to both effectively capture the relationships between input (images) and output (classes) as well as to to generalize to novel classes is non-trivial.

Without knowledge of the test classes, it is beneficial to exploit the discriminative capacity of attributes as a means of knowledge transfer. Hence, we propose an end-to-end deep neural network that predicts instances of unseen test classes based upon their visual attributes. We constrain (regularize) each layer of the network so as to encourage generalizability between the seen (training) and unseen classes, resulting in a model that we show to be more *transferable*. More specifically, our method learns a multi-layer nonlinear transformation from seen to unseen classes in a manner that constrains the individual layers in order to promote feature transferability between seen and unseen classes. We evaluate our method on four benchmark ZSL datasets and demonstrate results that comparable or exceed the performance of current state-of-the-art methods. Furthermore, we explore the use of word embeddings trained on large amounts of external text as additional semantic knowledge to further enhance transferability. We build class embeddings based on a weighted average of attribute embeddings that are built from a combination of word embeddings from the attribute description. These embeddings further improve performance for datasets with description-based attributes. As part of the evaluation, we also perform a series of ablations that demonstrate the contributions of the different components of our model, including layer-specific regularization, nonlinearity, architecture depth, and word embeddings. In addition, we evaluate our method on the related few-shot learning task to better understand the ability to learn classifiers when there is some overlap between training and test.

We summarize the primary contributions of the paper as follows:

- We propose an end-to-end, deep neural network model with layer-specific regularization and attribute class word embeddings that improves transferability in a zero-shot learning setting.
- We demonstrate word embeddings as effective attribute and class representations.
- We perform ablations evaluating the importance of individual model components.
- We achieve state-of-the-art results on several benchmark datasets and our method achieves greater transferability than existing state-of-the-art methods.

3.1 Related Work

An effective solution to zero-shot learning is to use the set of available seen images to learn a mid-level semantic embedding that relates an image and its corresponding class. Given an unseen test image, the problem then becomes one of first predicting its mid-level semantic embedding and then using these predictions together with the learned relationships to infer the object's class.

Visual attributes have proven to be a commonly used semantic abstraction for zero-shot learning. Among the first to propose the use of visual attributes is Lampert et al. (2009) who describe a two-step method whereby they first learn to predict the attributes associated with a given image. At inference time, they then use these predictions to estimate the classes based upon their attribute signatures. Similarly, Farhadi et al. (2009) learn a combination of semantic and discriminative attributes that are able to generalize both within and across object classes. Palatucci et al. (2009) consider a related zero-shot learning problem in which they project fMRI scans corresponding to people thinking about words into a space of manually specified features that they then use to predict the words. The effectiveness of such attribute-based methods is inherently limited by the accuracy of the intermediate attribute predictions, and learning accurate attribute predictors is challenging due to attribute correlations that are difficult to model (Jayaraman et al., 2014). Recognizing this limitation, Jayaraman and Grauman (2014) propose a random forest method that explicitly accounts for the unreliability of attribute predictions.

Rather than take a two-step approach, Akata et al. (2013) propose an alternative unified method that treats attribute-based image classification as a label-embedding problem, whereby they learn a function that expresses the compatibility between an image and the class embeddings. Recognition for a given image then follows by selecting the class with the most compatible embedding. Akata et al. (2015) learn the compatibility function using a combination of manual (supervised) attributes and unsupervised embeddings derived from unlabeled text corpora. Romera-paredes and Torr (2015) also take an integrated approach that does not distinguish between training (attribute prediction) and inference (class estimation) stages. Instead, they learn a simple linear embedding from attribute and image instances to their corresponding class in the form of linear regression model. The model first transforms images features to an attribute space via a linear projection learned during training. It then relates this attribute representation to class labels according to knowledge of their attribute signatures at test time. Integral to the effectiveness of the framework is their careful selection of regularizers as a means of learning parameters that better generalize to unseen classes. Our method similarly maps input images directly to their corresponding class via an implicit learned relationship between images and visual attributes. In contrast to Romera-paredes and Torr (2015), our model employs a multi-layer nonlinear network to represent the relationship between input images, visual attributes, and class labels. We learn this relationship using a different loss function and (layer-wise) regularizer, which, enables our method to better generalize to unseen classes, leading to significantly better results.

An alternative to human-labeled visual attributes is to directly learn the set of semantic embedding vectors from text corpora in an unsupervised fashion, and to then map images into this embedding space (Frome et al., 2013; Socher et al., 2013a; Norouzi et al., 2014; Ba et al., 2015). This has the advantage that the problem of predicting the class of an image essentially becomes a nearest neighbor search against vectors in the embedding space. (Ba et al., 2015) introduce a deep model that map raw text (i.e. encyclopedia articles) and image pixels to a joint embedding space which then used as a set of classifier weights for an object recognition network. Even though it seems there is similarity with our proposed model, there are significant differences from type of attributes to modeling of the problem using deep network. More specifically, our model differs in that we feed the CNN into a multi-layer nonlinear feedforward network that then takes the attributes as input at the last layer. We use layer-specific regularizers to enforce our model's generalizability. Frome et al. (2013) take advantage of the availability of large unannotated text corpora and describe an approach that combines a neural language model and a deep neural network for object recognition to learn to map images into a semantic embedding space. Classification then follows by choosing the nearest label in this embedding space. Socher et al. (2013a) similarly employ a neural language model to learn a semantic word space from a large unsupervised corpus of text. They then learn to map images into this space such that they are close to their corresponding semantic word vectors. These embeddings can then be used to infer the class of seen and unseen images based upon the semantic word vectors. Meanwhile, Norouzi et al. (2014) use a pre-trained classifier to map images into a semantic embedding space through a convex combination of the class label embedding vectors. Specifically, they use the predictive probabilities for different training labels from each of the classifiers to weigh the combination of the label embeddings in the semantic space. Zhang and Saligrama (2015) propose a parameterized method that learns a semantic similarity embedding using seen classes and then maps attributes into this space. They learn a similarity function that then embeds the target domain into this space.

An issue that arises with regards to the learned projection from an image to the semantic embedding space is the so-called projection domain shift problem (Fu et al., 2014), whereby learned projections are biased to the training data. Fu et al. (2014) propose a method that alleviates this issue by learning a multi-view semantic space that correlates image embeddings with the semantic embeddings. Additionally, attribute-based methods may suffer from noisy information in the form of missing or incorrect annotations, which can negatively affect prediction. Rohrbach et al. (2013b) seek to mitigate these limitations by proposing a transductive learning method that uses attributes as a means of transferring information from seen to unseen classes. They then use a graph-based algorithm to learn the manifold structure that underlies the novel classes. Mensink et al. (2014) use web-based data to learn co-occurrence statistics that they then use to weigh a combination of known classifiers to define predictors for new classes.

One-shot learning (Fei-Fei et al., 2006; Torralba et al., 2011; Lake et al., 2011; Walter et al., 2012) considers a related problem, relaxing the zero-shot learning assumption that the training and test sets are disjoint. Few-shot learning then considers the problem of learning to predict an object's class from a small number of training examples. Similarly, domain adaptation considers the scenario in which there is a large number of labeled examples drawn from a source domain that can be used for training but very few examples available for the target (testing) domain. Much of the work in domain adaptation focuses on natural language tasks (Blitzer et al., 2007; Glorot et al., 2011), however the technique has also been applied to object recognition (Saenko et al., 2010).

3.2 Our Framework

Our approach performs zero-shot learning in an end-to-end fashion by relating images to their corresponding class via an intermediate attribute-based semantic space. The method learns a nonlinear mapping from the input image and attribute vectors to the output class using a multi-layer feedforward neural network that takes as input the available attribute vectors and the image. We do so in a way that encourages the features to be generic and transferable by enforcing layer-specific regularizers with a Gaussian prior over the model's parameters. Next, we first derive the general model and then describe in detail the deep multi-layered network architecture with layer-specific regularizers. We then describe an extension of our model that uses word embeddings to build attribute and class embeddings, in order to further improve the transferability of the semantic space.

3.2.1 Problem Formulation

Let $F^s = \{(x_1^s, y_1^s), ...(x_N^s, y_N^s)\}$ be the training data that consists of N seen images $x_i^s \in \mathbb{R}^D$ in an embedding space and their corresponding labels y_i^s, where y_i^s is a one-hot indicator vector of length equal to the number of training classes K^s. The set of all training images and their labels can be defined as matrices $X^s \in \mathbb{R}^{D \times N}$ and $Y^s \in \mathbb{R}^{K^s \times N}$, respectively. Additionally, the data includes an attribute matrix $A^s \in \mathbb{R}^{P \times K^s}$, where each column a_j^s is a binary- or real-valued vector of P attribute assignments for class j. Further, we assume a held-out set of M unseen image-class pairs $F^u = \{(x_1^u, y_1^u), ..., (x_M^u, y_M^u)\}$ and their corresponding attributes $A^u \in \mathbb{R}^{P \times K^u}$, where the set of K^u test classes is disjoint from the set of K^s training classes, i.e., $K^s \cap K^u = \emptyset$. We represent the test images and classes as matrices $X^u \in \mathbb{R}^{D \times M}$ and $Y^u \in \mathbb{R}^{K^u \times M}$.

Given the input image feature embeddings and their corresponding attribute vectors, our goal is to learn a deep functional mapping from input space to output space[1]

$$y_i = f(x_i, A; W), \tag{3.1}$$

[1] For simplicity, we omit the superscripts, but will add them back later.

where W is a parameter matrix, which we define such that this mapping is sufficiently generic to transfer class prediction from the seen to unseen data.

3.2.2 Probabilistic Formulation

As with much of the previous work in zero-shot learning, we cast class prediction as a regression problem and later extend our formulation to the deep network model. Given an image x_i, we are interested in learning the conditional distribution over the class y_i

$$p(y_i|x_i; W) = \mathcal{N}(y_i; W^\top x_i, I), \tag{3.2}$$

where $\mathcal{N}(y; \mu, \Sigma)$ denotes a Gaussian distribution over a random variable y with mean μ and covariance Σ and $W \in \mathbb{R}^{D\times K}$ is a parameter matrix. We impose a Gaussian prior over the parameters to capture the relationship between different classes

$$p(W) = q(W) \prod_{j}^{K} \mathcal{N}(w_j; 0_D, \epsilon^2 I_D), \tag{3.3}$$

where w_j is the j^{th} column of W, 0_D is a D-dimensional zero mean vector, I_D is a $D \times D$ identity matrix, and ϵ^2 is a variance parameter. The term $q(W)$ takes the form

$$q(W) = \mathcal{MN}_{D\times K}(W|0_{D\times K}, I_D \otimes \Omega^c), \tag{3.4}$$

where $\mathcal{MN}_{D\times K}(W|M, \Sigma \otimes \Omega)$ is a matrix-variate normal distribution with mean $M \in \mathbb{R}^{D\times K}$, row covariance matrix $\Sigma \in \mathbb{R}^{D\times D}$, and column covariance matrix $\Omega \in \mathbb{R}^{K\times K}$ (Gupta and Nagar, 1999). The term $\mathcal{MN}_{D\times K}(W|M, \Sigma \otimes \Omega)$ can be expressed as (Gupta and Nagar, 1999; Zhang and Yeung, 2010)

$$\frac{\exp(\frac{-1}{2}\mathrm{tr}(\Sigma^{-1}(W-M)\Omega^{-1}(W-M)^T))}{(2\pi)^{DK/2}|\Sigma|^{K/2}|\Omega|^{D/2}}. \tag{3.5}$$

The row covariance matrix I_D in the Eqn. 3.4 models the relationship between input features, while the column covariance matrix Ω^c captures the relationship among the columns of W and, in turn, the classes (Zhang and Yeung, 2010). Hence, the first term $q(W)$ in Eqn. 3.3 models the structure of W and the second penalizes separately the complexity of each column (Zhang and Yeung, 2010). Eqn. 3.3 then serves as a means of regularizing the parameter matrix.

The posterior distribution over the parameter matrix follows as

$$p(W|X,Y) \propto p(Y|X,W)\,p(W). \tag{3.6}$$

Substituting the expressions in Equations 3.2 and 3.3 and taking negative logarithm, we formulate the maximum a posteriori estimate of the parameter matrix as the following optimization problem (Zhang and Yeung, 2010)

$$\min_{W} \sum_{i} \sum_{j} \mathcal{L}(f(x_i; w_j), y_i) + \eta \, \mathrm{tr}(W \Omega^c W^\top) + \gamma \|W\|_F^2 \tag{3.7}$$

where $\mathcal{L}$ is the squared loss function, $\gamma = 1/\epsilon^2$, and $f = W^\top x_i$.[2] In addition, we include a new regularization parameter η that modulates the trace norm penalty.[3]

3.2.3 Deep Network with Layer-specific Regularizers

Up to this point, we posed zero-shot learning as a regression problem without accounting for the attributes as an intermediate semantic representation. Next, we describe a modification to the optimization (3.7) that incorporates attributes with a nonlinear deep network architecture. We structure this model with layer-specific regularizers that promote the model's transferability to unseen classes.

Our network (Fig. 3.1) takes as input an image and the available attribute matrix. As with other vision architectures (Vinyals et al., 2015; Xu et al., 2015a), we pass each image through a convolutional neural network (CNN), followed by a series of fully-connected layers. These D-dimensional feature vectors feed into a fully-connected multi-layer network

[2] However, we note that this expression holds for any function f.

[3] In the case of the MAP estimate, $\eta = 1$.

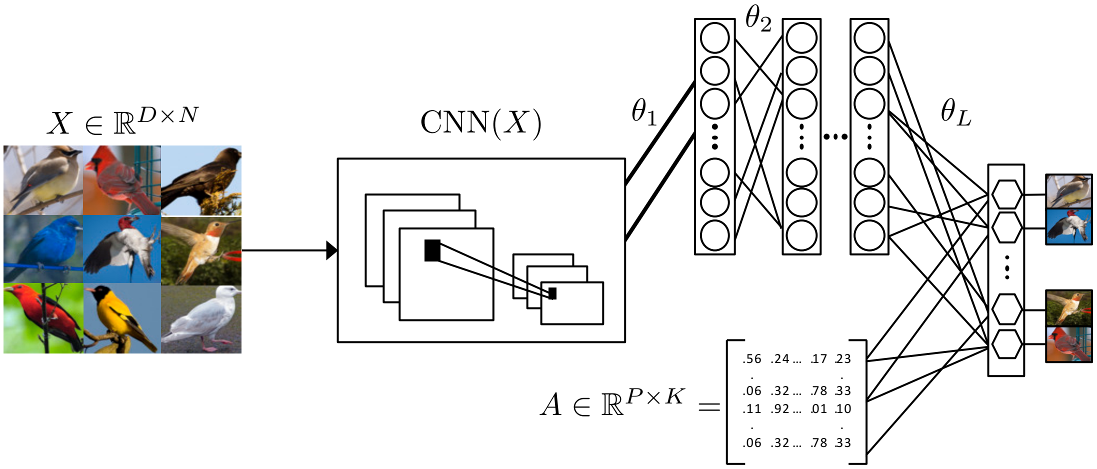


Figure 3.1. A visualization of our network architecture. A CNN embedding of input images is fed into a multi-layer network of which the last layer also takes as input the attribute matrix. The output is a prediction over the unseen class assignment..

on which we impose layer-specific regularization. This fully-connected network has the effect of representing the learned (during training) relationship between input images and an intermediate attribute representation. These learned features are fed into the last layer together with the attribute embeddings (at test time) to estimate the corresponding class label. In the results that we present shortly, we use the VGG-19 (Simonyan and Zisserman, 2014) and DeCAF (Donahue et al., 2014) networks as the CNN.

In our previous formulation, the columns of the parameter matrix W relate image embeddings with the object classes. We redefine the parameter matrix so as to use the attributes as an intermediate representation in order to better model the relationship between image embeddings and object classes. Specifically, we decompose the parameter matrix as $W = \Theta A$, where $\Theta \in \mathbb{R}^{D \times P}$. The term Θ maps the image embeddings into the space of attributes, while the attribute matrix A then relates this intermediate embedding representation to the space of object classes. Next, we move away from MAP estimation (Eqn. 3.7) and extend our optimization in order to express the complex nonlinear interactions between the input images, attributes, and classes. We do so by modeling the transformation Θ as a multi-layer network $g(\theta_l * g(\theta_{l-1}(g(..)..)))$, where g can be any activation function. In our architecture, we use rectified-linear units (ReLU) $g(X) = \max(0, X)$, which have proven effective at accelerating the convergence of stochastic gradient descent (Krizhevsky et al., 2012). As a result, with ReLU activation, we define the deep network architecture as

$$\max\Big[\,0, (...\max[\,0, X * \theta_1\,] * \theta_2...)\,\Big] * \theta_L, \tag{3.8}$$

where $\Theta = \{\theta_i\}_{i=1}^{L}$ with $\theta_i \in R^{D^i \times K^i}$ are the network parameters, D^i and K^i are number of parameters before and after layer i, respectively, and L is the number of layers. Furthermore, we update our architecture to then incorporate the attribute vectors as an additional input to the last layer of the network. A major advantage of adding the attribute vectors to only the last layer is that the other layers remain the same between training and test, since their parameter vectors θ_l do not depend on the number of classes. Similarly, the parameters of the last layer remain the same between training and test and the only change involves feeding in the new attribute matrix, i.e., $\theta_L A^s$ becomes $\theta_L A^u$.

The term tr$(W\Omega^c W^\top)$ in Eqn. 3.7 models the relationship between all classes based upon W and Ω. We modify this regularization to incorporate attributes as an intermediate embedding space between images and classes. In order to promote transferability in our model, we only apply this regularization to the last layer due to the fact that it tends to be more class-specific than the features close to the input, which tend to be more generic (Bengio et al., 2013). This has the additional benefit that it enables the network to focus on mapping the input to the output (class) space, while the last layer transfers knowledge between seen and unseen classes based upon the attribute embeddings. With this intuition, we introduce the term tr$(\theta_L A_s A_s^\top \theta_L^\top)$ for the last layer of our model. This term plays the same role as the tr$(W\Omega^c W^\top)$ term in the Eqn. 3.7, thereby favoring parameters θ_L that are less class-specific. In addition, we penalize the remaining layers with standard weight decay by penalizing the Frobenius norm. As a result, the optimization (3.7) can be rewritten as follows

$$\min_{\theta_l \in \Theta} \sum_{i}^{N} \mathcal{L}(f(x_i^s, A^s; \Theta), y_i^s) + \eta \, \mathrm{tr}(\theta_L A_s A_s^\top \theta_L^\top) + \gamma \sum_{l}^{L-1} \|\theta_l\|_F^2, \tag{3.9}$$

which defines the regularized optimization over the multi-layered network parameters. It is necessary to say that the novelty of our method is not solely in the use of the trace norm as a regularizer to impose structure, but rather in the way in which the trace norm is used only at the last layer in combination with the attribute vectors to allow transferability from seen to unseen classes. Further, our model uses different regularizers at the lower layers (i.e., $\gamma \sum_{l}^{L-1} \|\theta_l\|_F^2$) to control model complexity (i.e., avoid overfitting to seen classes). Together, these "layer-specific regularizers" enable our model to better generalize to new classes. The results demonstrate this characteristic as well as the contribution of our proposed model components (Table 3.1 and Table 3.2).

3.2.4 Learning and Inference

We train our network according to the optimization in Eqn. 3.9 using the hinge loss as the loss function

$$\mathcal{L}(f(x^s, A^s; \Theta), y^s) = \sum_{i} \sum_{j \neq y_i^s} \max(0, f(x_i^s, A_j^s; \Theta)_j - f(x_i^s, A_{y_i^s}^s; \Theta)_{y_i^s} + \Delta) \tag{3.10}$$

where Δ is a margin.[4] We train our model with stochastic gradient descent using the Adam algorithm (Kingma and Ba, 2015b). Having trained the model, we then perform inference over a given set of attributes A^u and an unseen test image x_i^u as

$$\arg\max_{j} f(x_i^u, A_j^u; \Theta), \tag{3.11}$$

where j specifies the class and f expresses the mapping from the image and attribute matrix to the output space, given the learned parameters Θ.

[4] In our experiments, we use a margin of $\Delta = 1$.

3.2.5 Attribute and Class Embeddings

In this section, we discuss further improvements to improve transferability by using word embeddings trained on large amounts of external data to build dense, continuous, and shareable attribute and class embeddings. Continuous word vector representations (Pennington et al., 2014; Mikolov et al., 2013; Faruqui et al., 2015) have recently proven very effective for various NLP (Turian et al., 2010; Collobert et al., 2011; Socher et al., 2013b; Bansal et al., 2014; Guo et al., 2014) and multimodal learning tasks (Frome et al., 2013; Norouzi et al., 2014) due to their ability to capture fine-grained semantic relationships between various words. Specifically, we use the Glove (Pennington et al., 2014) word vectors that combine the advantages of global matrix factorization and local context window methods.

We extend our model to use these dense word embeddings to better represent attributes and classes, which we later show results in improvements for domains that have descriptive attributes (e.g., *has_upper_tail_color::white*) rather than a single word. In doing so, we propose a new attribute representation based on word embeddings and the original real-valued attributes. In this new representation, we update the attribute-based class embedding (column) vector $a_i \in A$ associated with each class i as follows

$$a_i^{em} = [\frac{\sum_j^P a_i^j \cdot \text{Embed}(d^j)}{\sum_j^P a_i^j}], \tag{3.12}$$

where Embed(d^j) is a function that computes the embedding vector for the description d^j associated with the j^{th} attribute of class i. The attribute vector Embed(d^j) is built as the product of the Glove embeddings for each property word in the description attributes, e.g., a product of the word embeddings for *upper*, *tail*, and *white* in the case of the attribute description $d^j =$ *has_upper_tail_color::white* (leaving out the redundant word *color* repeated across such attributes). The original real attribute values a_i^j (provided as part of the data) are used as weights for the corresponding attribute's vector representation. This yields a class vector as a weighted average of the attribute vectors, themselves built from a product of the word embeddings in the attribute's description.

3.3 Experiments

We first analyze the overall performance of our model on the zero-shot learning task and subsequently present an ablation analysis to elucidate the contribution of the different components of our model. Next, we investigate the effects of word embeddings as an alternative semantic representation for the attribute-based class vectors. We then consider the effect of the ratio of seen-to-unseen classes on the overall accuracy as well as analyze the effectiveness of our method for the related task of few-shot learning. Note that the supplementary material provides further analyses that include evaluations of class confusion, precision and recall, and examples of correctly and incorrectly classified images.

3.3.1 Setup

3.3.1.0.1 Datasets We evaluate our method on four benchmark datasets: 1) SUN scene attributes (SUN) (Patterson and Hays, 2012) with 102 attributes, 707 seen classes, 10 unseen classes, and a total of $14,340$ images; 2) aPascal/aYahoo objects (aPY) (Farhadi et al., 2009) with 65 attributes, 20 seen classes, 12 unseen classes, and $15,339$ total images; 3) Animals with Attributes (AwA) (Lampert et al., 2009) with 85 attributes, 40 seen classes, 10 unseen classes, and a total of $30,475$ images; and 4) Caltech-UCSD Birds-200-2011 (CUB) (Wah et al., 2011) with 312 attributes, 150 seen classes, 50 unseen classes, and a total of $11,788$ images. The SUN, aPY, and CUB datasets include continuous, real-valued attributes, while AwA includes both real- and binary-valued attributes. In this paper, we only use real-valued attributes. We use publicly available 4096-dimensional DeCAF (Donahue et al., 2014) (AwA) features.[5] Additionally, we use VGG-19 (Simonyan and Zisser-

[5] `http://pub.ist.ac.at/~chl/ABC/`.

man, 2014) features made available Zhang and Saligrama (2015)[6] for the CUB, AwA, aPY, and SUN datasets.

Training Details While our method is amenable to training the entire network from scratch, we use pre-trained VGG-19 (Simonyan and Zisserman, 2014) and DeCAF (Donahue et al., 2014) networks for the CNN without fine-tuning in order to minimize memory consumption and to speed up training. We train our model on each dataset using the Adam gradient optimization algorithm (Kingma and Ba, 2015b) [7] We tune the hyper-parameters for our model, including the number of layers, the number of hidden units, the dropout rate, and settings for η and γ (Eqn. 3.9) on a validation set that consists of 20% of the class labels sampled from the training set. It is worth noting that there is no class overlap between the training and validation sets, which both follow the zero-shot setting. Having tuned the hyper-parameters, we re-train our model on the entire training set and report results on the test set. We use the inverted form of dropout (Srivastava et al., 2014) as an additional means of regularization, which does scaling at training time, in order to leave the forward pass during the test time untouched. We initialize the network parameters according to the procedures proposed by Glorot and Bengio (2010) and He et al. (2015). The training results in a three-layer network with 1000 hidden units for SUN, 1200 for aPY, and 800 for AwA; and a four-layer network with 1400 hidden units for the CUB dataset. Tuning resulted in parameter settings of[8] $\gamma = 14.48$ and $\eta = 0.29$ for AwA, $\gamma = 17.67$ and $\eta = 1.47$ for aPY, $\gamma = 15.81$ and $\eta = 10.17$ for SUN, and $\gamma = 17.29$ and $\eta = 0.01$ for CUB.

Table 3.1. Zero-shot learning accuracy percentage on all three datasets

Method	Image Feature	Attribute	AwA	aPY	SUN	CUB
Lampert et al. (2009, 2014)	Non-CNN	Real	41.40	19.10	52.50	-
Lampert et al. (2009, 2014)[i]	VGG-19	Real	57.23	38.16	72.00	-
Huang et al. (2015)	DeCAF	Binary	45.60	-	-	17.50
Jayaraman and Grauman (2014)	Non-CNN	Real	43.01 ± 0.07	26.02 ± 0.05	56.18 ± 0.27	-
Akata et al. (2013)	Non-CNN	Binary	43.50	-	-	18.00
Akata et al. (2015)	DeCAF	Real	61.90	-	-	40.30
Akata et al. (2015)	GOOGLE-1K	Real	73.90	-	-	**51.70**
Romera-paredes and Torr (2015)	Non-CNN	Binary	49.30 ± 0.21	27.27 ± 1.62	65.75 ± 0.51	-
Romera-paredes and Torr (2015)[ii]	DeCAF	Real	56.50 ± 0.012	-	-	-
Romera-paredes and Torr (2015)[iii]	VGG-19	Real	75.32 ± 2.28	24.22 ± 2.89	82.10 ± 0.32	-
Zhang and Saligrama (2015)	VGG-19	Real	76.33 ± 0.83	$\mathbf{46.23 \pm 0.53}$	82.50 ± 1.32	30.41 ± 0.20
Our method	DeCAF	Real	60.36	-	-	-
Our method	VGG-19	Real	**77.51**	**46.60**	**87.50**	47.56

[i] Reported elsewhere (Zhang and Saligrama, 2015). [ii] Our implementation. [iii] Reported elsewhere (Zhang and Saligrama, 2015).

3.3.2 Primary Results

We evaluate our method using the standard unseen and seen class splits for AwA and aPY (Lampert et al., 2014), the ten unseen class random split for SUN (Jayaraman and Grauman, 2014), and those used by Akata et al. (2013) for CUB.

Table 3.1 compares the performance of our method to that of several existing approaches in terms of multiclass accuracy, where numbers in bold denote the highest accuracy for each dataset. Our method achieves results that exceed existing state-of-the-art methods on the AwA and SUN benchmarks, and results are equivalent to the current state-of-the-art on aPY. Our method results in a relative improvement of 1.55% (1.18% absolute improvement) on AwA over that of Zhang and Saligrama (2015). On the aPY dataset, our method performs equivalently to their approach, while exceeding the other methods by a significant margin. On the SUN dataset, we achieve a relative improvement of 6.06%

[6] https://zimingzhang.files.wordpress.com/2014/10/cnn-features1.key.

[7] We use Adam (Kingma and Ba, 2015b) with learning rate of 0.0001 and settings of $\beta_1 = 0.9$ and $\beta_2 = 0.99$.

[8] Rounded to two decimal places.

(5.00% absolute improvement) relative to Zhang and Saligrama (2015), which is the previous state-of-the-art method. On the CUB dataset, our method lags behind that of Akata et al. (2015), which uses GOOGLE-1K image features, however it significantly outperforms all other methods.

3.3.3 Ablation Analysis

Next, we perform a series of ablation studies in order to show the contributions of the different components of our model, namely nonlinearity, multiple layers, and layer-specific regularization. Note that all methods include the Frobenius term for regularization and only ablate the trace term (Eqn. 3.9), which regularizes the last layer.

One ablation (denoted as L) corresponds to a simpler version of Eqn. 3.7 in which we perform classification with a linear mapping from input (images) to output (classes) without trace regularization. The second ablation (denoted as L+R) considers the same model, but with the inclusion of the trace regularization penalty, and corresponds to Eqn. 3.7. The next variation (denoted as NL) adds nonlinearity to the single-layer network (L). We then add multiple nonlinear layers (denoted as NL+D) to understand the contribution of a deep network. Finally, we consider our proposed model (denoted here as NL+D+R) that includes layer-specific regularization by adding back the trace term $\eta\ \mathrm{tr}(\theta_L A_s A_s^\top \theta_L^\top)$ in Eqn. 3.9.

Table 3.2. Analysis of our architecture components

Method	AwA	aPY	SUN	CUB
L	67.91	41.19	70.50	39.28
L+R	70.84	44.10	81.00	42.62
NL	76.36	40.62	81.50	45.76
NL+D	76.59	42.02	85.00	44.19
NL+D+R	**77.51**	**46.60**	**87.50**	**47.56**

Table 3.2 reports the ablation results across the four benchmark datasets. The inclusion of our regularization term results in a noticeable increase in classification accuracy. We also see an increase as a result of adding in nonlinearity, with the exception of aPY on which there is a slight decrease in accuracy. We see an additional boost in performance by adding additional layers. Overall, the ablation supports the claim that regularization is important to realizing a model that is able to transfer knowledge from seen to unseen classes.

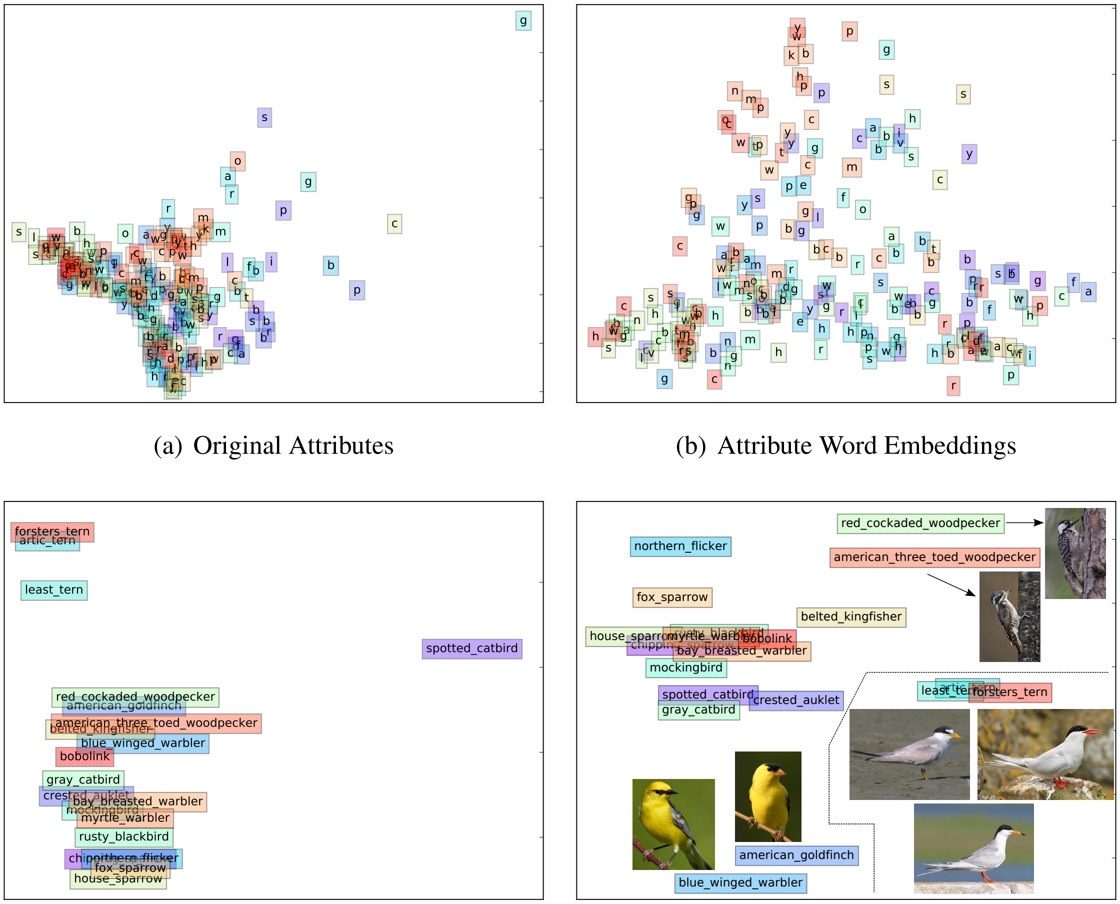
3.3.4 Attribute and Class Embeddings

Next, we evaluate the effectiveness of using pre-trained word embeddings as a richer, dense representation of the attributes in order to further promote generalizability. In our experiments, we use the 300-dimensional Glove Pennington et al. (2014) vectors. We evaluate the effect of word embeddings on the CUB dataset, since it contains more descriptive attributes than the other datasets (see examples and discussion in Section 3.2.5).

Table 3.3. Classification accuracy for with embeddings

Method	CUB
Original attributes	47.56
Embedding-Glove	$\mathbf{48.31}$

Table 3.3 presents the multiclass accuracy when using Embedding-Glove attribute-based class vectors, calculated per Eqn. 3.12. These attribute-based class vectors result in a relative improvement of 1.58% (0.75% absolute improvement) over our primary full model that uses the original attributes, hence enhancing further the results on CUB dataset. It is worth noting that it may be possible to further improve the contribution of word em-



(a) Original Attributes

(b) Attribute Word Embeddings

(c) Original Attributes (20 classes)

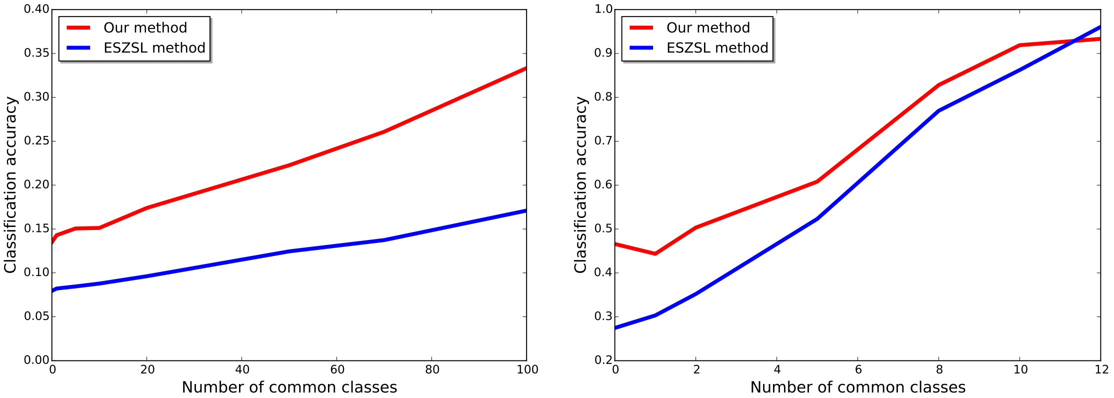
(d) Attribute Word Embeddings (20 classes)

Figure 3.2. Visualization of class embeddings for the CUB dataset..

beddings by fine-tuning the embedding vectors after initializing with the human-provided attributes. We leave this for future work.

3.3.5 Class Embedding Visualization

Next, we visualize the class embeddings for the CUB dataset and compare the original real-valued attribute-based vectors to those created using our method of weighted averaging of attribute word embeddings. In both cases, we use principle component analysis (PCA) and plot the top two dimensions. Figures 3.2(a) and 3.2(b) show the separation be-



(a) SUN Dataset

(b) aPY Dataset

Figure 3.3. Few-shot learning results on the (a) SUN and (b) aPY datasets..

tween the classes based upon the original vectors and the word embedding-based vectors, respectively, where we indicate each class by the first letter in its label. As one can see, the classes are well-separated using our embedding method, whereas they are all clustered closely together when using the original class vectors. This helps to explain the improvements in multiclass accuracy that we demonstrated in Table 3.3. Figures 3.2(c) and 3.2(d) then present the original and learned embeddings for a smaller set of 20 classes. As we can see, the "Red Cockaded Woodpecker" and the "American Three-Toed Woodpecker" (Fig. 3.2(d) top-right) are visually very similar and adjacent in our learned class embedding. Similarly, the "American Goldfinch" and the "Blue Winged Warbler" (Fig. 3.2(d) bottom-left) are also clustered in our embedding space and are similar in appearance. We see similar behavior for the "Last Tern," "Forster's Tern," and "Artic Tern" classes (Fig. 3.2(d) bottom-right), which share many similar visual attributes.

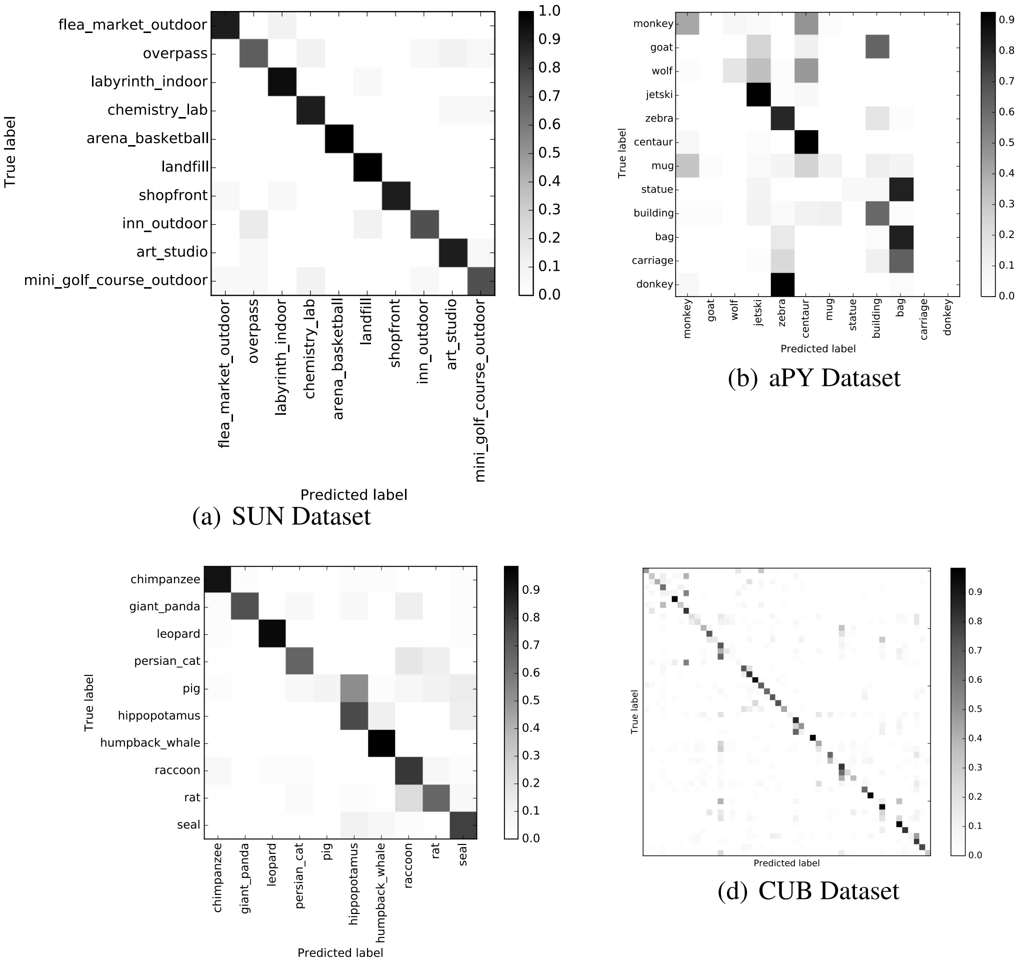
3.3.6 Few-shot Learning

We evaluate the performance of our model for the few-shot learning task in which one is allowed to see a limited number of classes from the test data during training. We do so using the SUN and aPY datasets. For SUN, we randomly select 317 classes as seen and 400 as unseen. We use a $40/60$ training/test split in order to have fewer images in training than in test. We use a standard split for aPY in which 20 classes are treated as seen and 12 are treated as unseen. It is worth noting that classes may be shared between training and test but not individual images.

Figure 3.3 plots the multiclass classification accuracy as a function of the number of classes shared between the seen and unseen sets. We start with the case of zero overlapping classes, which corresponds to the zero-shot scenario, and increase this number to 100 overlapping classes for SUN and 12 for aPY. We compare our approach to few-shot learning to the ESZSL method Romera-paredes and Torr (2015), and use the same setup and splits for both methods. As Figure 3.3(a) shows, our method outperforms ESZSL on SUN for all sample sizes in the few-shot learning setup. Figure 3.3(b) shows that our method yields greater performance on aPY when the number of shared classes is lower, but that the difference in accuracy decreases when there are more shared classes. We attribute the difference in behavior to the unbalanced nature of the aPY dataset.

3.3.7 Confusion Matrices

We first evaluate the multiclass classification accuracy by visualizing the confusion matrices for the four benchmark datasets. As can be seen in Figure 3.4, our method correctly discriminates between the classes on the AwA and SUN datasets. There is a higher rate of misclassification on the aPY and CUB datasets, which is consistent with the overall accuracies reported previously in Table 1. On the AwA dataset, we see that our method exhibits the most confusion between the “Hippopotamus” and “Pig” classes, which we attribute to the similarities in their appearance. The method also sometimes misclassifies “Persian Cat” images as being of the “Raccoon” class, the two of which are in fact visually similar in this particular set of classes.



(a) SUN Dataset

(b) aPY Dataset

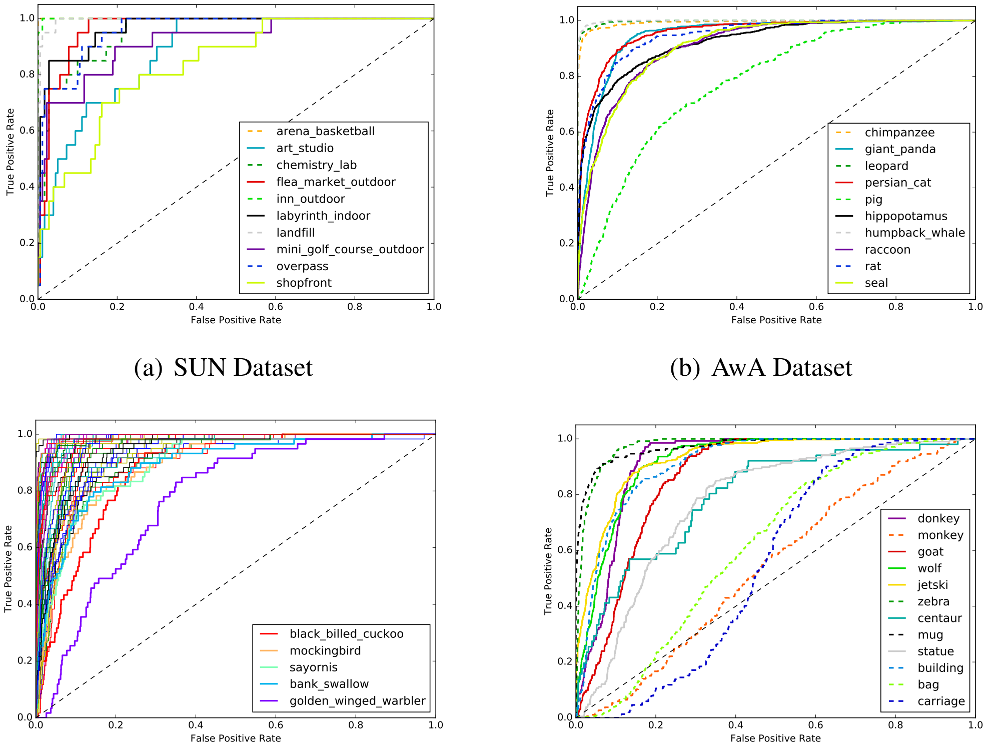
(c) AwA Dataset

(d) CUB Dataset

Figure 3.4. Confusion matrices for the four datasets that we consider. Note that we omit the labels for CUB due to lack of space..

3.3.8 ROC Curves

Next, we evaluate the multiclass accuracy by plotting the ROC curves for the different classes in each of the datasets, which we visualize in Figure 3.5. Our method learns accurate classifiers for all classes in the SUN (Fig. 3.5(a)) and AwA (Fig. 3.5(b)) datasets. Figure 3.5(c) presents the ROC curves for each of the 50 classes in the CUB dataset (where, for lack of space, the legend identifies the 5 classes with the lowest performance). Our method performs well on most classes, with one exception being the “Golden winged war-



(a) SUN Dataset

(b) AwA Dataset

(c) CUB Dataset

(d) aPY Dataset

Figure 3.5. Class-wise ROC curves for the (a) SUN, (c) CUB, (d) aPY, and (b) AwA datasets..

bler" class, which we discuss next. Meanwhile, our method has difficulty with a few of the classes within the aPY dataset (Fig. 3.5(d)).

3.3.9 Scalability Analysis

In order to explore the performance of our method when faced with imbalanced datasets with a bias against the number of seen classes, we consider a zero-shot learning scenario in which we vary the ratio of seen-to-unseen classes. The setup for this experiment follows that of Zhang and Saligrama Zhang and Saligrama (2015) who use the SUN dataset. First, we randomly select 17, 117, 217, and 317 out of 717 classes and individually consider them as seen classes. Second, we randomly select 10, 20, 30, and onwards

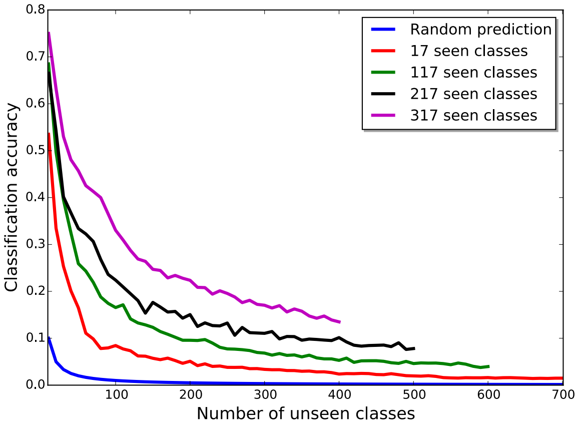
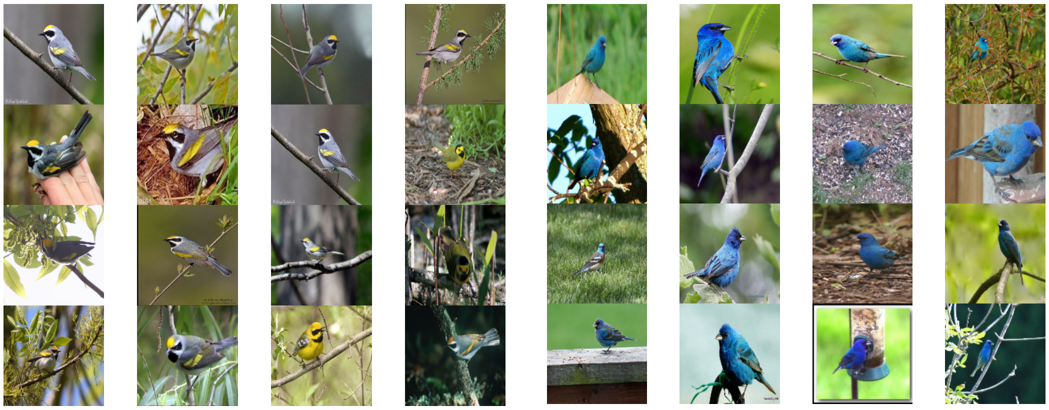


Figure 3.6. Accuracies for different ratios of seen-to-unseen classes..

from the remaining images (e.g., up to 400 in the case that we use 317 seen classes) as the unseen classes. For each of the combinations, e.g., 117 seen and 400 unseen, or 17 seen and 700 unseen, we train our network and report the classification accuracy. As shown in Figure 3.6, the performance degrades as the ratio of unseen to seen classes increases. In the case of 317 seen classes, the accuracy decreases by 55.96% when the number of unseen classes increases from 10 to 100, and decreases by 20.8% when going from 300 to 400 unseen classes. With 217 seen classes, the decreases in accuracy for these same ranges are 66.32% and 8.28%, respectively. Meanwhile, the accuracy decreases by 75.84% and 22.68%, respectively, when there are 117 seen classes. Finally, we see that with 17 seen classes, the accuracy decreases by 84.21% when the number of unseen classes increases from 10 to 100. However, the accuracy decreases by only 29.65% when the number of unseen classes increases from 300 to 400, and by only 16.66% when going from 400 to 500 unseen classes.

Overall, our model performs well when the number of unseen classes is small relative to the number of seen classes. Not surprisingly, the accuracies decrease as the number of unseen classes increase. This decrease is steep initially, but begins to slow down between 100 and 200 unseen classes.



(a) Failure Examples

(b) Success Examples

Figure 3.7. Images from (a) a class (“Golden winged warbler”) on which our method performs poorly as well as (b) a class (“Indigo bunting”) for which our method correctly classifies most images. In the latter, our method exploits the bird’s blue color as a discriminative attribute..

3.3.10 Success and Failure Examples

Finally, we provide examples from the CUB test dataset that demonstrate examples of classes that were easy as well as difficult for our model to classify. Figure 3.7(a) shows images from CUB for the “Golden winged warbler” class, which is one for which our method makes a large number of errors. Figure 3.7(b) shows example images of the “Indigo bunting” class on which our model makes only a few errors. Our method is able to exploit the discriminative attributes that exist this and other classes, such as the bird’s distinctive blue color.

3.4 Conclusion

We proposed a unified framework for zero-shot learning that relates input images with their corresponding class labels via an implicit embedding of visual attributes. Our model takes the form of a multi-layer nonlinear network that learns the relationship between images, attributes, and classes through layer-wise regularization. These layer-wise regularizers enhance the model's ability to learn features that generalize between seen and unseen classes. In this way, the model learns a nonlinear semantic projection that is able to transfer knowledge from the training space to the test space. Moreover, we propose the use of word embeddings as an alternative representation for attributes and classes, and show that these embeddings improve the performance of our model. Results on various benchmark datasets show that our method achieves greater transferability than existing state-of-the-art methods.

CHAPTER 4

Conclusion

In this dissertation, we first introduced an end-to-end memory-based attention model to describe an input video using natural language description. This model utilizes memories of past attention when reasoning about where to attend to in the current time step. This allows the model to not only reason about local attention more effectively, but also allows it to consider the entire sequence of video frames while generating each word. The experiments have confirmed that the memory component in the architecture plays a significant role in improving the performance of the entire network. It is worth noting even though in this work the problem of video caption generation has been considered, this model can be applied to any sequence learning problem, which remains as future works. As an another contribution, we introduced a simple but effective end-to-end deep network for zero-shot learning. The proposed method learns a nonlinear semantic projection that can be used to transfer knowledge from the training space to the test space. Results on several benchmark datasets demonstrate that this method achieves greater transferability than existing state-of-the-art methods.

Bibliography

R. Fakoor, A. rahman Mohamed, M. Mitchell, S. B. Kang, and P. Kohli, “Memory-augmented attention modelling for videos,” in *arXiv:1611.02261*, 2016.

R. Fakoor, M. Bansal, and M. Walter, “Deep transferable zero-shot learning with layer-specific regularizers and embeddings,” 2016.

R. Fakoor, M. Bansal, and M. R. Walter, “Deep attribute-based zero-shot learning with layer-specific regularizers,” in *NIPS 2015, Transfer and Multi-Task Learning Workshop*, 2015.

G. A. Sigurdsson, G. Varol, X. Wang, A. Farhadi, I. Laptev, and A. Gupta, “Hollywood in homes: Crowdsourcing data collection for activity understanding,” in *ECCV*, 2016.

S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, and K. Saenko, “Sequence to sequence – video to text,” in *ICCV*, 2015.

A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *NIPS*, 2012, pp. 1097–1105.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” in *NIPS*, 2014, pp. 3104–3112.

H. Fang, S. Gupta, F. Iandola, R. K. Srivastava, L. Deng, P. Dollar, J. Gao, X. He, M. Mitchell, J. C. Platt, C. Lawrence Zitnick, and G. Zweig, “From captions to visual concepts and back,” in *CVPR*, June 2015.

J. Devlin, H. Cheng, H. Fang, S. Gupta, L. Deng, X. He, G. Zweig, and M. Mitchell, "Language models for image captioning: The quirks and what works," in *ACL-IJCNLP*, July 2015, pp. 100–105.

K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *ICML-15*, 2015, pp. 2048–2057.

J. Johnson, A. Karpathy, and L. Fei-Fei, "Densecap: Fully convolutional localization networks for dense captioning," in *CVPR*, 2016.

O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *CVPR*, June 2015.

J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *CVPR*, 2015.

T. Rocktäschel, E. Grefenstette, K. M. Hermann, T. Kociský, and P. Blunsom, "Reasoning about entailment with neural attention," in *ICLR*, 2016.

Z. Yang, Y. Yuan, Y. Wu, R. Salakhutdinov, and W. W. Cohen, "Encode, review, and decode: Reviewer module for caption generation," *CoRR*, vol. abs/1605.07912, 2016.

D. L. Chen and W. B. Dolan, "Collecting highly parallel data for paraphrase evaluation," in *ACL*, Portland, OR, June 2011.

M. Rohrbach, W. Qiu, I. Titov, S. Thater, M. Pinkal, and B. Schiele, "Translating video content to natural language descriptions," in *ICCV*, Dec 2013, pp. 433–440.

S. Venugopalan, H. Xu, J. Donahue, M. Rohrbach, R. Mooney, and K. Saenko, "Translating videos to natural language using deep recurrent neural networks," in *NAACL HLT*, 2015.

L. Yao, A. Torabi, K. Cho, N. Ballas, C. Pal, H. Larochelle, and A. Courville, “Describing videos by exploiting temporal structure,” in *ICCV*, 2015.

P. Pan, Z. Xu, Y. Yang, F. Wu, and Y. Zhuang, “Hierarchical recurrent neural encoder for video representation with application to captioning,” in *CVPR*, June 2016.

T. H. Andrew Shin, Katsunori Ohnishi, “Beyond caption to narrative: Video captioning with multiple sentences,” *ICIP*, 2016.

Y. Pan, T. Mei, T. Yao, H. Li, and Y. Rui, “Jointly modeling embedding and translation to bridge video and language,” *CVPR*, 2016.

R. Xu, C. Xiong, W. Chen, and J. J. Corso, “Jointly modeling deep video and compositional text to bridge vision and language in a unified framework,” in *AAAI*, 2015.

N. Ballas, L. Yao, C. Pal, and A. C. Courville, “Delving deeper into convolutional networks for learning video representations,” in *ICLR*, 2016.

H. Yu, J. Wang, Z. Huang, Y. Yang, and W. Xu, “Video paragraph captioning using hierarchical recurrent neural networks,” in *CVPR*, June 2016.

D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” *ICLR*, 2015.

S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

J. Weston, S. Chopra, and A. Bordes, “Memory networks,” *CoRR*, vol. abs/1410.3916, 2014.

S. Sukhbaatar, A. Szlam, J. Weston, and R. Fergus, “End-to-end memory networks,” *NIPS*, 2015.

S. Sharma, R. Kiros, and R. Salakhutdinov, “Action recognition using visual attention,” *CoRR*, vol. abs/1511.04119, 2015.

A. Graves and N. Jaitly, “Towards end-to-end speech recognition with recurrent neural networks,” in *ICML-14*, 2014, pp. 1764–1772.

T. Brox and J. Malik, “Large displacement optical flow: Descriptor matching in variational motion estimation,” *TPAMI*, vol. 33, no. 3, pp. 500–513, March 2011.

K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *CoRR*, vol. abs/1409.1556, 2014.

K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *CVPR*, June 2016.

R. Pascanu, T. Mikolov, and Y. Bengio, “On the difficulty of training recurrent neural networks.” *ICML-13*, vol. 28, pp. 1310–1318, 2013.

D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *ICLR*, 2015.

J. Bergstra and Y. Bengio, “Random search for hyper-parameter optimization,” *JMLR*, vol. 13, pp. 281–305, 2012.

T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, “High accuracy optical flow estimation based on a theory for warping,” in *ECCV*, 2004.

D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, “Learning spatiotemporal features with 3d convolutional networks,” in *ICCV*, 2015.

S. Guadarrama, N. Krishnamoorthy, G. Malkarnenkar, S. Venugopalan, R. Mooney, T. Darrell, and K. Saenko, “Youtube2text: Recognizing and describing arbitrary activities using semantic hierarchies and zero-shot recognition,” in *ICCV*, 2013, pp. 2712–2719.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *CVPR*, 2009.

C. H. Lampert, H. Nickisch, and S. Harmeling, "Learning to detect unseen object classes by between-class attribute transfer," in *CVPR*, 2009, pp. 951–958.

C. Lampert, H. Nickisch, and S. Harmeling, "Attribute-based classification for zero-shot visual object categorization," in *TPAMI*, vol. 36, no. 3, 2014, pp. 453–465.

B. Romera-paredes and P. Torr, "An embarrassingly simple approach to zero-shot learning," in *ICML*, vol. 37, 2015.

S. Huang, M. Elhoseiny, A. Elgammal, and D. Yang, "Learning hypergraph-regularized attribute predictors," in *CVPR*, 2015.

Z. Zhang and V. Saligrama, "Zero-shot learning via semantic similarity embedding," in *ICCV*, 2015.

Z. Akata, S. Reed, D. Walter, H. Lee, and B. Schiele, "Evaluation of output embeddings for fine-grained image classification," in *CVPR*, 2015.

A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth, "Describing objects by their attributes," in *CVPR*, 2009.

M. Palatucci, D. Pomerleau, G. E. Hinton, and T. M. Mitchell, "Zero-shot learning with semantic output codes," in *NIPS*, 2009, pp. 1410–1418.

D. Jayaraman, F. Sha, and K. Grauman, "Decorrelating semantic visual attributes by resisting the urge to share," in *CVPR*, 2014.

D. Jayaraman and K. Grauman, "Zero-shot recognition with unreliable attributes," in *NIPS*, 2014.

Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid, “Label-embedding for attribute-based classification,” in *CVPR*, 2013.

A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, M. Ranzato, and T. Mikolov, “Devise: A deep visual-semantic embedding model,” in *NIPS*, 2013, pp. 1–11.

R. Socher, M. Ganjoo, C. D. Manning, and A. Ng, “Zero-shot learning through cross-modal transfer,” in *NIPS*, 2013, pp. 935–943.

M. Norouzi, T. Mikolov, S. Bengio, Y. Singer, J. Shlens, A. Frome, G. S. Corrado, and J. Dean, “Zero-shot learning by convex combination of semantic embeddings,” in *ICLR*, 2014, pp. 1–9.

J. Ba, K. Swersky, S. Fidler, and R. Salakhutdinov, “Predicting deep zero-shot convolutional neural networks using textual descriptions,” in *ICCV*, 2015.

Y. Fu, T. M. Hospedales, T. Xiang, Z. Fu, and S. Gong, “Transductive multi-view embedding for zero-shot recognition and annotation,” in *ECCV*, 2014, pp. 584–599.

M. Rohrbach, S. Ebert, and B. Schiele, “Transfer learning in a transductive setting,” in *NIPS*, 2013, pp. 1–9.

T. Mensink, E. Gavves, and C. G. M. Snoek, “COSTA: Co-occurrence statistics for zero-shot classification,” in *CVPR*, 2014.

L. Fei-Fei, R. Fergus, and P. Perona, “One-shot learning of object categories,” in *TPAMI*, vol. 28, no. 4, April 2006, pp. 594–611.

A. Torralba, J. B. Tenenbaum, and R. R. Salakhutdinov, “Learning to learn with compound HD models,” in *NIPS*, 2011.

B. M. Lake, R. Salakhutdinov, J. Gross, and J. B. Tenenbaum, “One shot learning for simple visual concepts,” in *Proc. Annual Conf. of the Cognitive Science Society (CogSci)*, 2011.

M. R. Walter, Y. Friedman, M. Antone, and S. Teller, “One-shot visual appearance learning for mobile manipulation,” in *IJRR*, vol. 31, no. 4, April 2012, pp. 554–567.

J. Blitzer, M. Dredze, and F. Pereira, “Biographies, Bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification,” in *ACL*, 2007.

X. Glorot, A. Bordes, and Y. Bengio, “Domain adaptation for large-scale sentiment classification: A deep learning approach,” in *ICML*, 2011.

K. Saenko, B. Kulis, M. Fritz, and T. Darrell, “Adapting visual category models to new domains,” in *ECCV*, 2010.

A. Gupta and D. Nagar, *Matrix Variate Distributions*, ser. PMS Series. Addison-Wesley Longman, 1999.

Y. Zhang and D. Yeung, “A convex formulation for learning task relationships in multi-task learning,” in *UAI*, 2010.

J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, “DeCAF: A deep convolutional activation feature for generic visual recognition,” in *ICML*, 2014.

Y. Bengio, A. Courville, and P. Vincent, “Representation learning: A review and new perspectives,” in *TPAMI*, vol. 35, no. 8, Aug 2013, pp. 1798–1828.

D. P. Kingma and J. L. Ba, “Adam: A method for stochastic optimization,” in *ICLR*, 2015.

J. Pennington, R. Socher, and C. Manning, “Glove: Global vectors for word representation,” in *EMNLP*, 2014, pp. 1532–1543.

T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in *NIPS*, 2013, pp. 3111–3119.

M. Faruqui, J. Dodge, S. K. Jauhar, C. Dyer, E. Hovy, and N. A. Smith, “Retrofitting word vectors to semantic lexicons,” in *Proceedings of NAACL*, 2015.

J. Turian, L. Ratinov, and Y. Bengio, “Word representations: A simple and general method for semisupervised learning,” in *ACL*, 2010.

R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, “Natural language processing (almost) from scratch,” in *JMLR*, 2011.

R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, “Recursive deep models for semantic compositionality over a sentiment treebank,” in *EMNLP*, 2013.

M. Bansal, K. Gimpel, and K. Livescu, “Tailoring continuous word representations for dependency parsing,” in *ACL*, 2014.

J. Guo, W. Che, H. Wang, and T. Liu, “Revisiting embedding features for simple semi-supervised learning,” in *EMNLP*, 2014.

G. Patterson and J. Hays, “Sun attribute database: Discovering, annotating, and recognizing scene attributes,” in *CVPR*, 2012.

C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, “The caltech-ucsd birds-200-2011 dataset,” California Institute of Technology, Tech. Rep. CNS-TR-2011-001, 2011.

N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” in *JMLR*, vol. 15, no. 1, 2014, pp. 1929–1958.

X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in *AISTATS*, 2010.

K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification,” in *ICCV*, 2015, pp. 1–11.