The Thesis Committee for Christopher Edwin Crabtree certifies that this is the approved version of the following Thesis:

Optimizing Visual Grounding of Latent Representations of Speech from Distant Language Groups

APPROVED BY

SUPERVISING COMMITTEE:

David Harwath, Supervisor

Greg Durrett

Optimizing Visual Grounding of Latent Representations of Speech from Distant Language Groups

by

Christopher Edwin Crabtree

THESIS

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Master of Science in Computer Science

THE UNIVERSITY OF TEXAS AT AUSTIN

December 2021

This is dedicated to my soon to be wife Molly. Without her love and support this thesis would not have been possible.

Optimizing Visual Grounding of Latent Representations of Speech from Distant Language Groups

Christopher Edwin Crabtree, M.S.Comp.Sci
The University of Texas at Austin, 2021

Supervisor: David Harwath

Recent years have seen an increasing research interest into using multi-modal grounding techniques to bolster classic natural language processing (NLP) and automated speech recognition (ASR) tasks. Previous work by Harwath et al. [5], demonstrated that visual grounding approximately doubled their model's bilingual utterance retrieval performance and similarly image retrieval was substantially improved by adding an alignment objective between languages. However, there is still much we don't know about the exact mechanism by which grounding is used in modern neural network systems. In this work, we extend the line of research pioneered by Harwath et al. by exploring empirically several contrastive learning frameworks and objectives designed to align input from different *modalities* (i.e. visual and speech input). Our experiments indicate potential avenues for improvement over the current best performing loss objective through analysis of our top two performing loss functions. We also find that in our trilingual setting, cross-lingual learning objectives can be removed to both improve image retrieval performance and reduce hyperparameter complexity.

Table of Contents

List of Tables

Chapter 1

Introduction

Recent years have seen an increasing research interest into using multi-modal grounding techniques to bolster classic natural language processing (NLP) and automated speech recognition (ASR) tasks. ASR tasks, in particular, generally require require a large amount of hand engineering of many sub-components. These include, but are not limited to: acoustic, pronunciation, and language models. Furthermore, once an ASR system is built, at least using current technology, the highly tuned sub-components cannot be repurposed into submodules of ASR systems for other languages. Thus an entirely new system must be built for each new language.

Visual grounding offers the potential to reduce the data requirements for training these ASR systems. For example, Srinivasan et al. [26] found that when degrading the audio input (using word masking) their novel neural architecture was better able to recover the original spoken utterance when leveraging features from a visual modality.

Additionally, research by Harwath et al. [8, 7] showed that visual input can be leveraged to learn speech representations containing useful linguist properties such as phonemes *without* explicit annotations. This indicates that multimodal information sources, when properly aligned, may reduce the annotation burden and sample com-

plexity of ASR retrieval systems. In this work, we investigate methods to improve this alignment.

Alignment between the representations of multimodal input is often evaluated using retrieval tasks between the representations [6, 25, 19]. For our work, we interested in two forms of alignment: that between speech and image representations, and between the speech from different languages that carry the same semantic information. We therefore assess representation alignment through the evaluation of two corresponding retrieval tasks: *semantic speech-to-image retrieval* and *semantic cross-lingual utterance retrieval*. Concretely, the task of semantic speech-to-image retrieval is to encode a spoken utterance that describes an image into a single vector representation and retrieve the corresponding image from a pool of encoded candidate images (in our case the pool is the entire dataset of images). Since both the utterance and the image are encoded into the same vector space, successful retrieval generally involves first training encoders to align output based on semantically meaningful information shared between the images and utterances.

Likewise in semantic cross-lingual utterance retrieval, the task is to retrieve an encoded utterance in a target language containing the same underlying semantic information as an utterance in a given source language utterance. The utterances in semantic cross-lingual utterance retrieval are not explicit translations of any kind, they merely convey or describe the same high level information. In this work, image descriptions are used to accomplish this. The dataset we use [22] contains 100,000 images with three descriptions of each image, with each description being in one of three distinct languages: English, Hindi, or Japanese.

As the utterances are not translations, lexical and syntactic structures in each language may be incredibly loosely aligned making low level co-occurrence information difficult to obtain. This creates a challenging learning environment for the representation encoders, but the hope is that the grounded visual information shared between the utterances will allow for a higher level semantic alignment. In fact, previous work by Harwath et al. [5] showed that semantic cross-lingual utterance retrieval performance (as measured by recall at 1,5, and 10) approximately double when including grounded visual information in the alignment optimization objective.

Aligning multi-lingual and multi-modal representations in such a setting presents significant challenges. Chief among them is the fact that there are still many aspects of end-to-end neural ASR models, that are not well understood, especially in multi-lingual settings. In particular, the exact alignment mechanism in modern neural models is difficult pinpoint so it is unclear what learning objective provides the best target for optimization. Furthermore, when fully optimizing jointly trained multi-lingual models, modern contrastive loss functions require an $O(n^2)$ number of pairwise comparisons. This complexity may be unnecessary in the presence of visual information and a simplification of the loss computation would reduce the total number of hyperparameters to tune. Finally, despite the success of parameter sharing systems in multi-lingual ASR systems, such as the work by Johnson et al.[17], systems with dedicated language encoders for each language still tend to outperform systems with shared encoders [3]. However, the additional of a visual modality may narrow this gap and encourage more parameter sharing among models when the shared conceptual recognition work is grounded in mutual sensory input.

In this work, we therfore attempt to address three research questions (RQs):

1. What type of learning objective (i.e. loss function) results in the best alignment of image and multi-lingual utterances as measured by average retrieval scores?

2. Aligning multiple modalities in the same representation space suffers a quadratic increase in loss terms with the pairing each new modality, which can complicate optimization. To what extent can this be mitigated?

3. When visual grounding information is present, are shared speech encoders an effective alternative to distinct language encoders?

The remainder of this thesis is structured as follows: the next section, 1.1, reviews related work, Chapter 2 discusses background and concepts relevant to understanding out main experiments, Chapter 3 presents our experiments aimed at answering RQ 1, Chapter 4 and Chapter 5, likewise, address RQ 2 and 3 respectively, and we conclude our thesis in Chapter 6.

1.1 Related Work

There are many areas of NLP interested in the potential of alignment with external modalities such as vision to enhance linguistic representations. Among these have been efforts to use information from visual modalities such as images to improve the performance of machine translation (MT) and automated speech recognition (ASR) models [14, 26, 5, 22]. These research efforts are often motivated by the intuition that humans most often learn language through processes that involve

resolving relations and co-occurrences between multiple simultaneous sensory input. Therefore, it is a reasonable hypothesis to assert that the addition of contextual information from modalities such as vision would be likely to aid in many natural language understanding tasks.

However, Calgayan et al. [1] noted that research efforts in multimodal machine translation (which aims to use auxillary input such as images to improve translation) have not yet clearly shown that that the addition of co-occurring visual information substantially improves MT performance. Their work did show, however, that when linguistic information is scarce or noisy, models using visual information are better able to recover from the missing information during translation. Calgayan et al. posited that the visual modality, therefore, was more useful for the purpose of regularization than as in important additional source of information for their models.

Similarly in the ASR domain, Srinivasan et al. [26] found that the visual modality served a similar purpose as that noted by [1] in MT. Namely, that visual information did not necessarily improve performance of speech recognition on clean datasets, but when degrading the audio input (using word masking) their novel neural architecture was better able to recover the original spoken utterance when leveraging features from a visual modality. These empirical observations seem to suggest that, contrary to intuition, additional information (in the form of co-occurring modalities carrying mutual semantic information) does not necessarily substantially improve a neural models' ability to recognize and utilize the patterns present in the linguistic modality.

However, Huang et al. [14] showed that in unsupervised MT settings they were

able to incorporate the visual modality to consistently improve translation without aligned corpora for training. Additionally, in Johnson et al.'s pioneering work in multilingual machine translation (translating between more than just two languages) [17], they showed that the addition of new language pairs *did* tend to improve transnational performance of their models. This phenomena has been termed transnational knowledge transfer [3] and aligns, in a very generally sense, with the intuition that richer information sources should enable pattern recognition algorithms, such as neural networks, to more quickly identify the salient variations in the input that are semantically relevant. This line of reasoning is further bolstered by results by Harwath et al. [5] which showed that the addition of visual context (in the form of pre-trained image features) approximately doubled the recall scores of cross-lingual audio retrieval between Hindi and English image descriptions. Cumulatively, these results indicate that latent visual representations *are* able to, in some fashion, improve alignments of the internal representations of sentences in different languages.

These conflicting results underscore the lack of current understanding of the mechanism by which neural models incorporate multimodal information. It remains unclear the extent to which visual grounding might aid in MT and speech translation (ST), as well as what might be the best way to incorporate visual information into these tasks. This lack of clear understanding motivates this Thesis's empirical exploration of multi-modal representation and in particular the alignment between semantically related multi-lingual speech utterances.

Chapter 2

Background

This section will review the primary concepts that will be evaluated empirically through the experiments in Chapters 3, 4, and 5

2.1 Task Description

All experiments in this work are designed to estimate predictive performance on retrieval tasks. This section formalizes our retrieval task and describes it in detail for the sake of clarity. We also define terms that well be used for the rest of the paper. Readers familiar with multi-modal and multi-lingual retrieval tasks are encouraged only to note the terminology we define in this section.

We are given a $\underline{\mathcal{D}}$ataset, $\mathcal{D}$, containing a set of K distinct information sources. We will refer to each information source as a $\underline{\mathcal{V}}$*iew*. Section 2.3 discusses this convention further. We will refer to an arbitrary view as $\mathcal{V}_j$ where $j \in [1, K]$ indexes each view. In our dataset, each view contains N samples and each sample, $\nu_{ij} \in \mathcal{V}_j, i \in [1, N], j \in [1, K]$ has a corresponding entry in each of the other $K - 1$ views which share some sort of semantic relatedness. Furthermore, each $\mathcal{V}_j$ has it's own representation scheme for it's constituent samples making a conventional tensor representation with fixed dimensions for all samples in $\mathcal{D}$ inexact. This gives the

generalized dataset for our task the following definition:

$$\mathcal{D} = \{(\nu_{i1}, \nu_{i2}, \cdots, \nu_{iK}) : \nu_{i1} \in \mathcal{V}_1, \nu_{i2} \in \mathcal{V}_2, \cdots, \nu_{iK} \in \mathcal{V}_K, \quad i \in [1, N]\}$$

To reduce ambiguity, we will refer to an arbitrary K-tuple in $\mathcal{D}$ as a *datapoint* and an arbitrary member of a K-tuple/datapoint, ν_{ij}, as a *viewpoint*.

To be more concrete for a moment, in the dataset used for this work we have $N = 100,000$ with $K = 4$ views. These views are: 1) a set of images, 2) English descriptions of each image, 3) Hindi descriptions of each image, and 4) Japanese descriptions of each image (although we do not assume that particular ordering). For each image in the image view, there is a corresponding spoken utterance in each of the three language views describing that image. An image and the three corresponding language utterances make up a datapoint and are each individually a viewpoint.

Also note that, on occasion, we further categorize our views based on *modality*. To be clear, this dataset contains two modalities: vision and audio/speech; one view is in the vision modality (images) and three views are in the audio modality (English, Japanese, and Hindi utterances). We therefore commonly differentiate retrieval results based on modality pairing, with retrieval results from vision-audio pairs being referred to as 'semantic image-to-speech retrieval' (or simply 'image retrieval') and retrieval results for audio-audio view pairs referred to as 'semantic speech-to-speech retrieval' (or 'cross-lingual retrieval' for short). Section 3.1.3 discusses our naming conventions in more detail. This partitioning is made to enable readers to more clearly assess the models' ability to capture the types of mutual information shared between two modalities.

As an aside, *mutual information* (MI) has a well-known and well-defined mathematical form, but we will be using the term in the idiomatic sense. Recent work by Tschannen et al. [28] has shown that many popular lower-bound maximization schemes for estimating MI break down in practice and do not correlate with expected performance gains. Furthermore, MI is only well-studied in the two variable (view) case. The generalization of MI to three or more variables is referred to as *interaction information* and has certain properties (such as permitting negative values) that make it difficult to interpret. For this reason we eschew MI's technical definition and use it in a more intuitive sense.

Keeping these points in mind, we still wish to encode the datapoints from each view into a shared vector space and maximize the alignment between mutually associated viewpoints (by way of retrieval maximization). Our task, then, in words is: given 1) an arbitrary viewpoint, $\nu_{ij} \in (\nu_{i1}, \nu_{i2}, \cdots, \nu_{ik})$, from j^{th} view and the i^{th} datapoint in $\mathcal{D}$, and 2) the set of viewpoints in a target view $\mathcal{V}_t$, retrieve the view point $\nu_{it} \in \mathcal{V}_t$ that is in the same datapoint as ν_{ij}.

In mathematical form, we define a 'success' in this task as:

$$\mathbb{1}_{\{i\}}[retrieve(\mathcal{V}_t, \nu_{ij})] \tag{2.1}$$

Where $retrieve(\mathcal{V}_t, \nu_{ij})$ is some function which returns the index of the viewpoint in $\mathcal{V}_t$ which it predicts is associated with ν_{ij}. Also, here $\mathbb{1}_A(x)$ is a general indicator function that evaluates to 1 if $x \in A$ and 0 otherwise.

In order to implement $retrieve(\mathcal{V}_t, \nu_{ij})$ we first use *encoder* functions, f_{θ_t} and f_{θ_j}, to produce encoded representations of the viewpoints in the target view and

ν_{ij} (respectively) in the same vector space, $\mathbb{R}^d$. d is a parameter defined by the encoders. In this work all encoders take the form of neural networks, and in particular convolutional neural networks (CNNs).

With all viewpoints encoded into the same vector space, we can now more easily calculate a *similarity score* through which we can rank the similarity between $f_{\theta_j}(\nu_{ij})$ and all $f_{\theta_t}(\nu_{lt}),\ \nu_{lt} \in \mathcal{V}_t$. This similarity score is measured by a chosen *similarity function*, S. S follows the typical mathematical definition of a similarity measure. Namely, that it is the inverse of a distance metric which follows the usual distance metric constraints: it takes two arguments, it is always non-negative, it is symmetric in it's arguments, it obeys the triangle inequality, etc. The similarity score can then be is written as:

$$s_{i,l} = S(f_{\theta_j}(\nu_{i,j}), f_{\theta_t}(\nu_{l,t})) \tag{2.2}$$

The retrieval is then performed by computing all $s_{i,l},\ l \in [1, N]$ and returning the index, l_{max}, corresponding to the viewpoint in the target view that achieves the highest similarity score with ν_{ij}. With these terms defined, we can refine our definition of a 'successful' retrieval in Equation 2.1 as follows:

$$\mathbb{1}_{\{i\}}\ [\arg\max_{l \in [1,N]} S(f_{\theta_j}(\nu_{i,j}), f_{\theta_t}(\nu_{l,t}))] \tag{2.3}$$

In fact, one of the most common metrics for evaluating retrieval models, *recall at 1*, is computed by applying Equation 2.3 across all viewpoints in V_j, summing the results, and normalizing by N. Intuitively, this gives the percentage of successful retrievals of viewpoints in $\mathcal{V}_t$ (using viewpoints of $\mathcal{V}_j$ as input). *Recall at k* is computed similarly,

with only minor alterations to Equation 2.3 to more leniently define success as i being included in the top k indices as ranked by S.

A natural learning objective, then, for the parameters of f_{θ_j} and f_{θ_t} might then be to maximize the number of successful retrievals in the dataset $\mathcal{D}$. This can be written as:

$$\max_{\theta}\Big(\sum_{(\nu_i 1,\cdots,\nu_i k)\in\mathcal{D}}\sum_{j=1}^{K}\sum_{t>j}^{K}\mathbb{1}_{\{i\}}\left(\arg\max_{l\in[1,N]} S(f_{\theta_j}(\nu_{i,j}), f_{\theta_t}(\nu_{l,t}))\right)\Big) \tag{2.4}$$

Unfortunately, optimizing this objective directly is typically intractable, though, and must be approximated. The approximation commonly takes the form of using an alternate loss function designed to: 1) maximize the similarity scores of correctly paired viewpoints (likewise minimizing incorrect pairs' scores), and 2) allow for easier optimization. Most often, the alternate loss is chosen to be a differentiable loss function and optimized using gradient-descent based optimization algorithms.

There are a number of loss-functions in common use for this type of maximization objective, many of which are motivated by maximizing the estimate of the aforementioned MI estimate. Which of these loss functions produces optimal results for our particular setting is the subject of the experiments in Chapter 3. In the following section we describe the losses relevant to our experiments in detail.

2.2 Loss Functions

There are numerous loss functions that are designed to encourage similarity (w.r.t. a defined similarity function) among pairs or groups of datapoints. We explore a subset of these in this work an describe the important aspects of each below.

2.2.1 Triplet Loss

Triplet loss has been used extensively in the literature to learn distance metrics since gaining prominence through the work such as that of Schultz et al. [24] Likewise, Triplet loss has perhaps the longest history of use of any loss objective in for our particular task. Harwath et al. [6] first introduced the task of semantic speech-to-image retrieval and used a maximum margin objective between each final unpooled image embedding and final sequence embedding of paired images and utterances. This objective eventually became formulated as the triplet loss.

Generally, triplet losses take an *anchor* embedding/vector A, a *truthy* embedding T (which is positively associated with the anchor), and a *falsey* embedding F, also known as an imposter, that is not associated with the anchor. Then, given a chosen similarity function, S, the triplet loss can be calculated as:

$$Triplet(A, T, F) = \max(0, S(A, F) - S(A, T) + M) \tag{2.5}$$

Where M is some margin to be chosen as a hyperparameter (often just 1). This loss function, when used with gradient descent algorithms, effectively pushes the A and T embeddings toward each other and the A and F embeddings away from each other. The margin, then, influences how strongly model is encouraged to push or pull the associated embeddings.

This was formalized by Harwath et al. [10] for spoken image retrieval by using two triplet losses: one defined with image representations as anchors, and one with

spoken captions as anchors. For a single datapoint it can be written as:

$$\begin{aligned} T(I_i, C_i, I_i^{imp}, C_i^{imp}) &= \Big(\max(0, S(I_i, C_i^{imp}) - S(I_i, C_i) + 1) \\ &\qquad + \max(0, S(C_i, I_i^{imp}) - S(C_i, I_i) + 1) \Big) \\ &= Triplet(I_i, C_i, C_i^{imp}) + Triplet(C_i, I_i, I_i^{imp}) \end{aligned} \tag{2.6}$$

Where S is a chosen similarity function. I_i and C_i are the i^{th} image/caption which are trade off as both the anchor and truthy embeddings. I_i^{imp} and C_i^{imp} are the i^{th} *imposter* samples of an image and caption not associated with the correct image/caption pair. In this formulation, the first max function term is effectively the triplet loss from Equation 2.5 when I_i is viewed as the anchor, C_i is viewed as T, and C_i^{imp} is viewed as F. The second max term then uses C_i as the anchor and the images in likewise fashion. The imposters are sampled from a uniform distribution from within the minibatch.

This form of triplet loss is highly dependent on the imposter samples chosen. This is because in the final stages of optimization, most negative samples will already be far from the anchor. If imposters are then chosen poorly the optimization may progress very slowly. This motivates hard and semi-hard negative sampling [16] which, generally speaking, seek to sample imposter/negative examples that are very close to the anchor (and there for 'hard' to distinguish from the positives). This general approach was adapted to the spoken image retrieval task by Harwath et al. [9] which showed that the use of semi-hard negatives increased retrieval performance.

The accompanying loss function, was formulated more generally in [8] as:

$$\mathcal{L}_{trip}(\theta) = \sum_{i=1}^{N} T(I_i, C_i, I_i^{imp}, C_i^{imp}) + T((I_i, C_i, \tilde{I}_i^{imp}, \tilde{C}_i^{imp}) \tag{2.7}$$

Where T is the same as in Equation 2.6 and N is the size of the dataset, but can of course be implemented with standard minibatching techniques. The important difference between the terms is that $\tilde{I}_i^{imp}$ and $\tilde{C}_i^{imp}$ are chosen to be the image and caption that are most similar to their respective anchors in the batch (as measured by the similarity function S), instead of the uniform distribution as I_i^{imp} and C_i^{imp}

2.2.2 InfoNCE and Masked Margin Softmax (MMS)

The Masked Margin Softmax (MMS) loss was introduced by Ilharco et al. [15] and uses many more negative examples for each positive pair than triplet loss. MMS loss effectively computes the categorical cross-entropy loss between an anchor embedding and K other embeddings, only one of which is the positive corresponding embedding.

Ilharco et al. define MMS loss as follow: let B equal the batch size and X and Y be the final encoded representations of B samples of two arbitrary information sources (i.e. views) The MMS minibatch can then be written as:

$$MMS(X, Y) = \frac{1}{B} \sum_{i=1}^{B} \frac{e^{S(X_i, Y_i) - M}}{e^{S(X_i, Y_i) - M} + \sum_{k=1}^{B-1} \mathbb{I}[k \neq i \wedge k \notin \mathcal{Z}] e^{S(X_i, Y_k)}} \tag{2.8}$$

Where $\mathcal{Z}$ is a set of predefined indices corresponding to examples to be *masked* (hence the name) and disregarded from the loss calculation. In our dataset there are no examples of this sort so $\mathcal{Z}$ is always empty.

Notice that $MMS(X, Y)$ is not symmetric. Specifically, each representation X_j is contrasted with all other Y_k in the batch, but Y_j is not given the same treatment. The full MMS loss for a batch completes the symmetry and is defined as:

$$\mathcal{L}_{MMS}(\theta) = MMS(X, Y) + MMS(Y, X) \tag{2.9}$$

The margin M is an important quantity and can be a constant value, updated according to a schedule, or adaptively updated according to some function of the batch. When $M = 0$ this loss has the form of what is often referred to in the literature as InfoNCE loss [23]. Several works use this terminology [11, 13], but there seems to be a lack of consensus over this issue [27]. Still, we hesitantly follow the current conventions and whenever an experiment uses a margin of 0, we will refer to it as InfoNCE.

The margin can also be set to a constant, set on a predefined schedule as in Ilharco et al. [15], or adaptively updating using heuristics derived from the batch such as in Monfort et al. [21]. Each of these methods are explored empirically in Section 3.1.3.

2.2.3 Hypersphere Loss

The hypersphere loss was introduced by Wang [30] and attempts to reframe conventional contrastive loss through the notions of *alignment* and *uniformity* of latent representations on the hypersphere. They provide empirical evidence that directly optimizing their measures of alignment and uniformity can outperform contrastive losses. Given a batch of size B with X and Y defined in the same manner as

in Section 2.2.2, the *alignment* sub-loss is defined as:

$$HSphere_{align}(X, Y) = \frac{1}{B} \sum_{i=1}^{B} \|X_i - Y_i\|_2^{\alpha}$$

where α is a hyperparameter to be tuned.

The *uniformity* sub-loss is calculated as:

$$HSphere_{uniformity}(X, Y) = \frac{1}{B(B+1)/2} \sum_{i=1}^{B} \sum_{j \geq i}^{B} e^{-t\|X_i - Y_j\|_2^2}$$

with t being another hyperparameter. With the sub-losses defined the full hypersphere loss is defined as:

$$\mathcal{L}_{hsphere}(\theta) = \lambda_a * HSphere_{align}(X, Y) + \lambda_u * HSphere_{uniformity}(X, Y) \tag{2.10}$$

With λ_a and λ_u being additional weighting parameters.

In the original work, Wang et al.'s experiments showed that values of λ_a and λ_u around 1 tended to work well, but λ_u often needed to be lower than λ_a. Also, in the original formulation of the hypersphere loss an additional base contrastive sub-loss term was added to Equation 2.10 with an accompanying weighting hyperparameter, but they found that it was not necessary for optimal performance so we have omitted it in our experiments for simplicity.

2.3 Multiview Contrasting Frameworks

Work by Tian et al. [27], provided a framework for using contrastive loss functions with several *views*. They define a view as a particular sensory input during a single event. In this way, there may be multiple views all describing the same

underlying (or latent) event. A view, then, can have a more abstract meaning in which some amount of information contained in the view is shared among other views.

They then present training schemes to maximize the latent information shared by all views. The two designs relevant to our experiments are the *full-graph* and *anchor* training schemes.

Both schemes start with a symmetric contrastive loss function, $\mathcal{L}_{sym}(V_j, V_t)$, defined over two batched views, $V_j \in \mathcal{V}_j$ and $V_t \in \mathcal{V}_t$, where V_j and V_t are both in $\mathbb{R}^{d \times B}$. B is again the batch size and d being the size of each representation vector. Each element in V_j are assumed to have a corresponding positively associated element in V_t that carries some mutual information about an event of interest (and vice-versa for the samples in V_t). We will assume as well that the positive examples share the same index in the batch. Note that here V_j and V_t refer to the output of encoders f_{θ_j} and f_{θ_t}, not the raw input. To simplify notation we omit the encoder notation in this section.

The contrastive loss function $\mathcal{L}_{sym}$ is also is also assumed to be symmetric in it's arguments. It should encourage alignment between the positive examples in two views while discouraging alignment between the examples that do not carry mutual information (again, alignment w.r.t. some similarity function defined by $\mathcal{L}_{sym}$). Notice that the loss functions defined in Equations 2.10, 2.7, and 2.9 all meet the above assumptions.

Finally, the full-graph and anchor frameworks attempt to maximize the mutual

information contained in a set of views $\mathcal{V}_i, i \in [1, K]$.

With the above assumptions and notation, the full-graph framework computes the following loss for each batch:

$$\mathcal{L}_{graph}(\theta) = \frac{1}{K(K+1)/2} \sum_{j=1}^{K} \sum_{t \geq j}^{K} \mathcal{L}_{sym}(V_j, V_t)$$

In the full-graph framework, the encoders for each view are encouraged to find vector representations that are all mutually close to one another.

The anchor framework, on the other hand, is defined by having all contrasting operations involve a single 'anchor' view. It is calculated as:

$$\mathcal{L}_{anchor}(\theta, i) = \frac{1}{K-1} \sum_{t \neq j} \mathcal{L}_{sym}(V_j, V_t)$$

Where i denotes the index of the anchor view. With the anchor framework, mutual information that may be shared among views outside the anchor are disregarded. This permits a somewhat looser alignment representation, since the only requirement is to have each view be 'similar' to the anchor, but not necessarily each other.

2.4 Neural Architectures

We use convolutional neural networks (CNN) as our base architecture. The focus of this work is on the effect of loss functions and training regime in a multilingual image retrieval and as such we do not attempt to optimize architecture. We therefore use two off the shelf encoders for our images and speech utterances making minimal changes to the default settings of each. Each encoder produces several representations for each input, though, and our loss functions require a single vector

representation. This section briefly discusses the image and audio encoders and then describes strategies for output pooling.

2.4.1 Image Model

The image encoder is based on the ResNet-50 model proposed by He et al. [12], which is a 50 layer deep CNN. It is pretrained on ImageNet classification, but the final layer has been altered slightly. We use the same image pre-processing as in Harwath et al. [8]. We also remove the final softmax and fully connected layer and instead perform a 1x1 convolution to obtain a desired size of the final image representations as was done in [8]. The final model size is over 25 million parameters and the output representation for each image is a 7x7x1024 tensor.

2.4.2 Audio Model

We will first describe the pre-processing steps taken on the audio and then briefly describe the architecture.

2.4.2.1 Audio Pre-Processing

The same pre-processing is used for all experiments. Log-Mel filter bank (LMFB) spectrogram features are calculated from raw audio with 40 frequency bins. We use Hamming-windowed frames with a 25 ms width and shift of 10 ms. During batching, we pad the variable length utterances to the longest utterance in the batch. However, to reduce the amount of padding, before training begins we truncate the longest utterances in each language by identifying the longest 5% of utterances and

truncating them to the length of the longest utterance outside of the 5%.

2.4.2.2 Architecture

We use the audio/speech model presented by Harwath et al.[8]. It is a 17 layer CNN with the first being a 1D convolution of 128 filters of size 1x40x1 across the time dimension. This effectively transforms and expands the 40 LMFB featurs of each time step into 128 features defined by each filter. The following 16 layers are divided into 4 residual *speech* blocks. Each speech block starts with an initial 1D convolution layer with kernel size of 1x9 with a stride of two that serves to downsample the input, followed by a batch norm layer. This followed by three successive 1D convolution (kernel size 1x9, stride 1) and batch norm layers. Additionally, a residual connection is is added between the input and output of the with a 1x1 convolution to align dimensions when necessary. No padding is used by the convolutional layers. The final model size is over 46 million parameters and produces 1024 dimensional embeddings of various length depending on the batch.

2.4.3 Output Pooling

As noted in the previous section, the output of each view encoder do not necessarily align across all dimensions. This creates a problem for most distance/similarity measures which typically require vector representations. Work by [9] explored a more complex set of similarity measures which utilize similarity measures across the time and location dimension, but for this work we restrict ourselves to a simplified setting of similarity measures defined only on two vectors of the same dimension. This is also

more realistic in terms of retrieval when the set of target viewpoints is very large. We therefore require some sort of pooling mechanism before similarity scores can be calculated and retrieval can be performed.

Experiments in section 3.3 compare pooling strategy performance empirically, but the following sections describe the basics of each mechanism.

2.4.3.1 Average Pooling

This is the simplest pooling strategy and consists of first flattening all dimensions before the last, then averaging across the flattened dimension and maintaining the embedded representation dimension. One complication is the padding added to resolve the variability of sequence length in the speech encoders. Using a simple average would diminish the features of the meaningful parts of shorter sequences. Also, the sequence length is reduced in each layer due the convolutional downsampling, which means there is likely a point in the final output sequence that corresponds to both padded and unpadded input.

To deal with this we only average the first $avg\_pool\_len_{ijk}$ sequence embeddings of the k^{th} utterance in the i^{th} batch of the j^{th} view. We denote len_{ijk} as the original sequence length of the utterance. We calculate avg_pool_len as:

$$avg\_pool\_len = \lfloor len_{ijk} / \text{round}\Big(\frac{max\_sl_{ij}}{max\_sl\_outij}\Big) \rfloor$$

Where max_sl_{ij} is defined as the sequence length of the longest utterance in in the input of the j^{th} language view in the i^{th} batch and $max\_sl\_out_{ij}$ is the sequence length of that corresponding utterance after encoding. For our dataset, $max\_sl\_out_{ij}$

typically ranges from 50 to 500. round$(\cdots)$ rounds to the nearest integer. Here $\frac{max\_sl_{ij}}{max\_sl\_outij}$ can be thought of as the reduction ratio from the input sequence length to the output sequence length.

2.4.3.2 Multi-Head Self-Attention Pooling

Considering the simplistic nature of average pooling, we also experiment with using a self attention layer to perform the pooling. Previous work by Ohishi et al. [22] found that a single headed self-attention layer was beneficial when used *before* the pooling layer (specifically between the second to last and last convolutional layers), but they still used average pooling for their final representation.

Inspired by this result and the recent proliferation and success of Transformer layers [29], which uses multi-headed self-attention (MHSA), we attempt to use this as an adaptive pooling mechanism for our final representation. Specifically, we use 8 heads and an positional encoder layer and we chose the largest $max\_sl\_out\_ij$ to be the maximum sequence length for the positional encoder. After the positional encoding, we apply a dropout layer to impose a layer specific hyperparameter we can use to counteract the potential for MHSA to overfit to the dataset.

Another implementation note is that we chose to prepend an additional learnable embedding to each flattened/sequence dimension to act in the same capacity as the CLS token used in BERT encoders [4]. We then use the first embedding (which corresponds to the position of prepended embedding) as our final representation. We also experimented with simply using the first embedding of the sequence and the results of this are discussed in section 3.3

As a final note, we use sequence masking to prevent the final representation from attending to the embeddings corresponding to padding. For this layer we used the PyTorch implementation of MHSA in our experiments and it added approximately four million parameters to each encoder.

2.4.3.3 Transformer Pooling

We also experiment with using a full Transformer block using a single embedding to serve as the final representation. We used the same settings and adaptations as described in 2.4.3.2 regarding number of heads, positional encoding, prepending a learned embedding, and masking. We chose to use 2048 as the internal feed forward dimension, which is the same as the original Transformer. Together this layer accounted for approximately an additional 8 million parameters for each encoder over average pooling.

Chapter 3

Exploring Learining Objectives

This chapter explores the extent to which the overall learning objective effects retrieval performance, indicating better alignment of each views vector representations. We note that the terms 'learning objective' and 'loss function' are used in this chapter interchangeably.

3.1 Experimental Design

This section describes the procedures used for out data preparation, training, and evaluation. We also describe variations to our model's output that we will explore. The base architecture, though, is unchanged from the description in Section 2.4.

3.1.1 Data

The experiments in this chapter (and all proceeding chapters) use a dataset consisting of 100,000 images from the Places 205 dataset [31] that have additional English, Hindi [5], and Japanese [22] spoken descriptions corresponding to each image. It is important to note that the descriptions from each language are not translations of each other as each speaker is only directed to describe the image and has no knowledge of what was said previously. A thorough description of the characteristics of the each

languages utterances can be found in Ohishi et al.[22]. We use a training set of 99,000 datapoints and use the testing set of 1,000 datapoints defined by Harwath et al. [5] for evaluation.

3.1.2 Training

We use the Adam optimization algorithm [20] in all of the following experiments with an exponential decay rate of .9 for the gradient and .999 for the squared gradient. We also use a batch size of 128 and a learning rate of .001 with a scheduler and train each model for 75 epochs. This scheduler enforces a linear warmup schedule (from 0 to .001) through the first 10% of training steps, and then decreases by a factor of .99 after every 50 steps.

In this chapter's experiments, all training is done using the full-graph framework of multi-view training as described in Section 2.3. This is to simplify experimental variables, such as choosing which view will be the anchor, as well as to give the encoders easy access to all information shared between the views. The Chapter 4 will explore ways to reduce the complexity of full pairwise comparisons.

Importantly, all loss functions under consideration have some component designed to encourage dissimilitude. Without this, predictions can find a trivial local solution by collapsing on constant value for all representations. Contrastive loss functions use 'negative examples' to serve this role where as loss functions like the hypersphere loss contain the uniformity objective. In either case, we supply the non-positive examples in each batch to these sub-components. To be clear, all loss functions draw their negative examples from within their respective batch.

3.1.3 Testing and Reporting Metrics

During inference, for each datapoint, each viewpoint is measured with respect to a similarity function with every viewpoint in the other views. We use the dot product as our similarity function. However, since the hypersphere loss is optimized for representations with unit length, we similarly scale the output before retrieval. This can also be thought of as using cosine similarity as our similarity function for retrieval.

Recall at 1, 5, and 10 are then calculated by checking if the correct target viewpoint ranked in the top 1, 5, or 10 most similar viewpoint in the target view. Calculation of these metrics are discussed in Section 2.1. Importantly, for each view pair we report only the average recall at k scores between each view pair and omit the scores for individual retrieval directions. Precisely, for each view pair, two sets of recall at k scores are calculated: 1) a set in which the viewpoints of the first view of the pair are treated as the source input and the second view is the target, and 2) a set where the second of the pair is treated as the source input and the first view is the target. These two directions do not necessarily produce the same results, but we did not observe substantial deviation from the average among any view pair in any experiment in this work. We then average the two scores to obtain the overall average recall at k. We report only the average in this manner because of space concerns and also because we are primarily interested in overall alignment in the latent space, not in any one particular retrieval direction.

Additionally, we distinguish between view pairs of different modalities in our reporting (i.e. image-speech pairings) and those within the same modality (speech-

speech). We will refer to average results from speech-speech retrieval scores as 'cross-lingual retrieval results' and, for simplicity, refer to the average results from speech-image pairings simply as 'image retrieval results'. We acknowledge, though, that retrieving speech utterances from image input as not 'image retrieval' an any sense, but use the phrase to simplify terminology.

3.1.4 Experimental Variables

We separately explore two experimental variables in this chapter: choice of loss function and output pooling mechanisms. The variations of each are outlined below.

3.1.4.1 Loss Functions

The loss functions under consideration were described in detail in Section 2.2, but we for completeness we list them here. We compare:

1. The hypersphere loss (Equation 2.10) with several values of λ_u.
2. The triplet loss (Equation 2.7).
3. InfoNCE (Equation 2.9 with $M = 0$).
4. Masked Margin Softmax, M=1 (Equation 2.9 and referred to as MMS in our experimental results).
5. Masked Margin Softmax, scheduled updates for M following the original work by Ilharco et al. [15]. This is labeled as MMS_Sch. in results.

6. Masked Margin Softmax, with adaptive updates as introduced by Monfort et al. [21], which is also called Adaptive Mean Margin loss (AMM), but we refer to as MMS_Adp. in our experiments.

3.1.4.2 Output Pooling

The output pooling layers were also discussed as length in Section 2.4.3. We state them here along with important hyperparameters to be explicitly:

1. Average pooling.
2. Multi-headed Self-Attention (MHSA) with varying values of learning rate and dropout for the layer. Final output representation is chosen from the first embedding in the sequence of the output of the MHSA layer.
3. Transformer layer with the final output chosen as with MHSA.

3.2 Learning Objective Experiments

This chapter's experiments involve six distinct loss functions: InfoNCE, hypersphere loss, triplet loss, masked margin softmax (MMS) loss, scheduled MMS loss, and adaptive MMS loss. We compare performance of each directly in Section 3.2.2. However, the hypersphere loss contains many unique hyperparameters. No similar loss function has ever been applied to this task so optimal hyperparameters have not previously been studied. We therefore describe our tuning procedure in Section 3.2.1.

After comparing loss functions in Section 3.2.2, we also experiment with different pooling strategies in Section 3.3. Some final settings for pooling layers are

Image Ret	E&I.R1	H&I.R1	J&I.R1	E&I.R5	H&I.R5	J&I.R5	E&I.R10	H&I.R10	J&I.R10	Ep
Unif. 1	0.10%	0.10%	0.05%	0.60%	0.45%	0.50%	1.20%	0.85%	1.35%	19
Unif. .875	0.10%	0.15%	0.05%	0.40%	0.75%	0.25%	0.95%	1.10%	0.75%	34
Unif. .75	**2.90%**	**3.15%**	**3.70%**	**12.65%**	**12.10%**	**15.20%**	**22.55%**	**20.95%**	**26.70%**	16
Unif. .5	0.10%	0.10%	0.10%	0.50%	0.50%	0.50%	1.00%	1.00%	1.00%	0

Cross-Ling	H&E.R1	J&E.R1	J&H.R1	H&E.R5	J&E.R5	J&H.R5	H&E.R10	J&E.R10	J&H.R10	Ep
Unif. 1	0.10%	0.10%	0.15%	1.00%	0.45%	0.45%	2.45%	0.85%	0.80%	19
Unif. .875	0.15%	0.10%	0.10%	0.80%	0.65%	0.55%	1.50%	1.30%	1.05%	34
Unif. .75	**1.75%**	**1.40%**	**2.05%**	**8.10%**	**10.75%**	**8.95%**	**13.95%**	**20.00%**	**15.80%**	16
Unif. .5	0.10%	0.10%	0.10%	0.50%	0.50%	0.50%	1.00%	1.00%	1.00%	0

Table 3.1: Hypersphere loss w/ Average pooling

Recall at 1, 5, and 10 results for the hypersphere loss (columns grouped by recall) when varying the uniformity weighting hyperparameter λ_u. View pairs separated by an & with image (I), English (E), Hindi (H), and Japanese (J) view pairs. The best scores in each row are bolded and the epoch in which these scores were achieved is included in the final column.

explored in Section 3.4.

3.2.1 Hypersphere Loss Tuning

As the hypersphere loss is markedly different than the loss functions previously explored on this dataset, we chose to tune the uniformity sub-loss weighting, λ_u. We chose this hyperparameter because it appeared to be the most impactful to the final results of the original paper [30]. There are in fact several hyperparameters available for adjustment for the hypersphere loss, but due to time constraints we were unable to tune the others. We chose an optimization scheme of starting with the baseline value (1.0), and decreasing by .25 until performance ceased to increase. We kept all other hyperparameters constant during this tuning. In particular, we held the alignment loss weighting, λ_a at 1.0, $t = 2$, and $\alpha = 2$. We used the encoder implementations described in 2.4 and used average pooling to combine the final output representations

The results for the all view pairings can be found in figure 3.1. As can be seen in both image retrieval and cross-lingual settings, the overall performance of the

Image Ret.	E&I.R1	H&I.R1	J&I.R1	E&I.R5	H&I.R5	J&I.R5	E&I.R10	H&I.R10	J&I.R10	Ep
Hyper	8.30%	7.85%	9.00%	25.20%	23.10%	30.55%	36.95%	33.75%	43.65%	16
Triplet	16.40%	12.95%	22.70%	39.20%	32.65%	53.15%	51.55%	43.55%	67.05%	36
InfoNCE	18.05%	14.70%	28.00%	41.55%	36.10%	59.20%	54.55%	45.85%	72.35%	26
MMS	12.65%	16.15%	25.50%	36.65%	36.45%	57.45%	50.30%	**47.60%**	71.50%	19
MMS_Sch.	**18.85%**	**16.50%**	**28.35%**	42.15%	**37.65%**	58.05%	54.90%	46.85%	71.95%	24
MMS_Adp.	16.90%	14.45%	27.30%	**45.25%**	36.90%	**61.05%**	**57.50%**	47.15%	**74.40%**	29
Cross-Ling	**H&E.R1**	**J&E.R1**	**J&H.R1**	**H&E.R5**	**J&E.R5**	**J&H.R5**	**H&E.R10**	**J&E.R10**	**J&H.R10**	**Ep**
Hyper	5.90%	6.90%	6.40%	18.00%	21.30%	19.30%	27.05%	33.95%	28.95%	16
Triplet	9.65%	8.05%	6.30%	23.35%	24.70%	21.80%	32.85%	35.85%	30.70%	36
InfoNCE	11.55%	11.40%	**10.50%**	27.25%	29.00%	**27.30%**	37.55%	41.30%	**36.80%**	26
MMS	8.35%	7.90%	8.85%	22.10%	25.00%	24.70%	31.80%	36.85%	35.60%	19
MMS_Sch.	**12.00%**	11.25%	9.40%	**28.45%**	**32.25%**	26.35%	**38.00%**	**44.60%**	36.60%	24
MMS_Adp.	10.65%	**12.25%**	10.15%	26.30%	32.00%	25.40%	36.00%	42.40%	35.70%	29

Table 3.2: Learning Objective Comparion

Recall at 1, 5, and 10 results of different learning objectives. View pairs separated by an & with image (I), English (E), Hindi (H), and Japanese (J) view pairs. Results are the average of recall scores when retrieving in both directions. Average pooling of output representations used in all cases.

hypersphere loss is highly dependent on the weighting of the uniformity sub-loss. In all following experiments, we use $\lambda_u = .75$ as it was the best performing uniformity weighting.

3.2.2 Loss Function Experiments

This set of experiments compares retrieval results when optimizing encoder parameters with respect to several loss functions. Specifically, we compare InfoNCE loss, triplet loss, MMS loss with a margin of 1, scheduled MMS loss, adaptive MMS loss, and hypersphere loss. Training, inference, and evaluation procedures are as described in sections 3.1.2 and 3.1.3.

Results for image retrieval pairings and cross-lingual results can be found in Table 3.2. Recall at 1, 5, and 10 are listed for each loss. We highlight several findings from this experiment:

1. For image retrieval, the top performers are the adaptive and scheduled MMS losses, with the scheduled MMS performing well for recall at 1 and adaptive MMS improving in higher recalls of 5 and 10.

2. For cross-lingual retrieval, several losses work well, but there is no clear standout among top performers.

3. Despite the simpler design, InfoNCE provides comparable results to scheduled and adaptive MMS losses.

Finding 1 suggests that for further optimization, some small tweaking may be possible to the adaptive MMS loss to encourage similar behavior to the hand tuned scheduled MMS loss during certain parts of the training.

Findings 1 and 2, though, generally align with that of Ohishi et al[22]. Their experiments showed that scheduled MMS loss outperformed hard negative triplet loss. Our results, bolster their results by providing evidence that it outperforms a broader range of loss functions and can perhaps be optimized slightly.

Finding 3 is important because unlike scheduled and adaptive MMS loss, our formulation of InfoNCE has no hyperparameters and requires no tuning. Because of this, all remaining experiments will use InfoNCE loss.

3.3 Pooling Experiments

As described in Section 2.4.3, there are many different strategies to pool the final output representations of each view. Since neither MHSA nor Transformer layers

Image Ret	E&I.R1	H&I.R1	J&I.R1	E&I.R5	H&I.R5	J&I.R5	E&I.R10	H&I.R10	J&I.R10	Ep
MH Attn.										
Lr:.001 D:.1	0.25%	0.05%	0.15%	0.70%	1.15%	1.10%	1.55%	1.75%	2.10%	1
Lr:.0001 D:.1	14.10%	9.70%	18.70%	34.20%	27.40%	46.85%	45.70%	37.70%	62.15%	15
Lr:.001 D:.3	11.05%	9.05%	19.10%	33.80%	25.35%	45.35%	44.85%	34.65%	59.45%	21
Lr:.001 D:.5	4.55%	3.65%	6.20%	14.15%	13.60%	20.85%	23.50%	21.25%	30.05%	32
Avg. Pool	**18.05%**	**14.70%**	**28.00%**	**41.55%**	**36.10%**	**59.20%**	**54.55%**	**45.85%**	**72.35%**	26
Cross-Ling	H&E.R1	J&E.R1	J&H.R1	H&E.R5	J&E.R5	J&H.R5	H&E.R10	J&E.R10	J&H.R10	Ep
MH Attn.										
Lr:.001 D:.1	0.20%	0.15%	0.20%	1.65%	0.70%	0.80%	2.70%	1.05%	1.45%	1
Lr:.0001 D:.1	7.60%	6.70%	5.75%	20.85%	19.80%	17.15%	29.20%	29.30%	26.50%	15
Lr:.001 D:.3	6.50%	6.25%	5.00%	18.80%	19.05%	15.65%	27.80%	26.90%	22.40%	21
Lr:.001 D:.5	2.20%	2.05%	1.85%	9.25%	7.95%	7.85%	14.80%	12.90%	12.60%	32
Avg. Pool	**11.55%**	**11.40%**	**10.50%**	**27.25%**	**29.00%**	**27.30%**	**37.55%**	**41.30%**	**36.80%**	26

Table 3.3: Multi-head self-attention pooling.

Results of varying learning rate and dropout values for multi-headed self-attention pooling. Results from average pooling are shown as a baseline.

have been used a pooling strategy in this task, we first attempt to tune hyperparameters of the simpler of the two layers, the MHSA. We chose to adjust the learning rate and the dropout percentage. Due to time constraints we were not able to fully tune these parameters, but we were able to explore an array of settings. The image retrieval cross-lingual results for these settings along with a baseline of average pooling can be found in Table 3.3. All models were trained using InfoNCE loss.

Comparing these results with those for average-pooled, MHSA is clearly underperforming. We first found that the original learning rate of .001 made learning unstable and decreasing to .0001 helped considerably. We next hypothesized that the lower performance might be due to some amount of overfitting of the attention mechanism. This prompted us to adjust the percentage of dropout applied after the positional encoding layer. Our results contradict this hypothesis as it seems increasing regularization did not improve generalization.

Overall, these results were surprising given the general success of MHSA as

well as the additional four million parameters (approx.) that were added to each encoder. Further investigation is necessary to explain these results.

3.4 Additional settings

The results from Section 3.3 prompted us to try other configurations of MHSA as well as the Transformer block. First, we tried to simply use the first embedding of the flattened sequence instead of the additional learned prepended token (resembling the CLS token in BERT). We also tried to place the MHSA layer between the second to last and last layer of the language encoders and simply feed that to the last convolutional layer to then get average-pooled. This is a similar strategy of that explored by [22], but we maintain our 8 head configuration rather than using a single head. The learning rate for this setting was returned to .001 (since that worked well previously for average pooling) and dropout the dropout kept at .1. Finally we try using the Transformer block for output pooling, which adds a layer norm layer and two linear layers to the MHSA. For the Transformer block, we used the best learning rate and dropout percentage from the MHSA experiments (.0001 and .1 respectively). All models are trained using the InfoNCE loss.

Results for image retrieval can be found in Table 3.4, we use the average pooling layer and tuned MHSA from Section 3.3 as baselines. The recall scores indicate that removing the prepended token has a marginally negative impact on performance. Somewhat surprisingly, though, the internal MHSA configuration and the Transformer block proved to substantially harm retrieval. This may be due to a lack of robust tuning, but it should be noted that average pooling has no hyperpa-

Image Ret	E&I.R1	H&I.R1	J&I.R1	E&I.R5	H&I.R5	J&I.R5	E&I.R10	H&I.R10	J&I.R10	Ep
MH:No CLS	9.55%	7.90%	16.55%	27.85%	23.75%	43.15%	38.50%	32.55%	57.50%	21
MH:Internal	1.15%	1.10%	3.40%	4.30%	3.90%	11.15%	7.90%	6.50%	18.85%	33
Transformer	0.30%	0.25%	0.35%	1.10%	0.90%	2.05%	2.25%	2.15%	3.35%	50
Base MHSA	14.10%	9.70%	18.70%	34.20%	27.40%	46.85%	45.70%	37.70%	62.15%	15
Avg. Pool	**18.05%**	**14.70%**	**28.00%**	**41.55%**	**36.10%**	**59.20%**	**54.55%**	**45.85%**	**72.35%**	26
Cross-Ling	H&E.R1	J&E.R1	J&H.R1	H&E.R5	J&E.R5	J&H.R5	H&E.R10	J&E.R10	J&H.R10	Ep
MH:No CLS	5.25%	4.20%	5.35%	15.95%	16.55%	17.60%	23.30%	24.80%	24.65%	21
MH:Internal	1.95%	0.50%	0.55%	6.70%	2.80%	2.45%	10.95%	4.90%	4.70%	33
Transformer	0.25%	0.10%	0.00%	0.95%	0.65%	0.60%	1.35%	1.25%	1.10%	50
Base MHSA	7.60%	6.70%	5.75%	20.85%	19.80%	17.15%	29.20%	29.30%	26.50%	15
Avg. Pool	**11.55%**	**11.40%**	**10.50%**	**27.25%**	**29.00%**	**27.30%**	**37.55%**	**41.30%**	**36.80%**	26

Table 3.4: Transformer Pooling and Additional Configurations for MHSA.
Results of removing the CLS token from the MHSA, using MHSA between the third and fourth language encoder layer (internal), and using a Transformer block. Baselines of the tuned MHSA and the average pooling layer are added for comparison.

rameters itself and does not require tuning. Due to this lack of requisite tuning and it's relatively strong performance, we use average-pooling for the experiments in the remaining chapters of this thesis.

3.5 Summary

In this chapter we compared a broad range of loss functions for the tasks of image and cross-lingual retrieval. Through our experiments we showed that Masked Margin Softmax-based losses with variable margins (updated using either a schedule or an adaptive updating rule), generally outperformed the other losses. InfoNCE, though, was a very strong competitor and contains no hyperparameters (for a single view pair) in our formulation, which make it an attractive option as well.

In Section 3.2.1 we also demonstrated the difficulties present in tuning loss-specific hyperparameters. This can become problematic when fully optimizing the alignment of $O(n^2)$ view-pairs, which may require view-pair specific weighting to

reach optimal performance. In Chapter 4.1 we explore the effects of reducing the complexity of view-pairs in the full loss function.

Chapter 4

Loss Complexity

State-of-the-art methods for aligning vector representations, currently require contrastive loss functions. Since most contrastive loss functions contrast between only two views at a time, a system designed to align multiple views in the same vector space requires a $O(n^2)$ number of comparisons between views. Since full optimization of such systems requires extensive hyperparameter tuning, this quadratic growth can result in an untenable number of parameters to tune. For example, work by Chen et al. [2] and He et al. [11] have noted that state-of-the-art contrastive loss objectives are sensitive to the number of negative examples. It is unclear currently how the increase in number of views/modalities effects this need for negative examples making it a hyperparameter that grows with the number of view pairs.

This chapter's experiments are motivated by reducing this complexity. We explore strategies to reduce the growth to a linear increase and assess the impact on performance. We start with our experimental design in Section 4.1, then we discuss our results in Section 4.2.

4.1 Experimental Design

Most aspects of our experimental setup are identical to Section 3.1, including the dataset and encoder architectures. Again each view has a dedicated encoder and final representations are obtained by average pooling. We use same Adam optimizer with a learning rate of .001 and InfoNCE loss.

4.1.0.1 Experimental Variable

Our experimental variable involves the multi-view training framework outlined in Section 2.3. Instead of using all view-pairs (i.e. the *full-graph*) as in Chapter 3, this chapters experiments all use the *anchor* framework. The experimental variable, then, is our choice of anchor. We use three different anchors in our experiments.

In the first we simply use the image view as the anchor. That is, we only compute the InfoNCE loss from view pairs that include the image view. As the only view in the visual modality, this seems a natural choice. This is referred to as the *image anchor* experiment.

In the second variation, we use an 'average view' as the anchor, which is an average representation of all views. Specifically, for each datapoint we average the final representations of each viewpoint (recall our definition of datapoint and viewpoint from Section 2.1). This effectively creates a centroid for each datapoint. Within the InfoNCE loss, our intuition is that positively associated viewpoints will be drawn closer to their own centroid, while being pushed away from centroids of other datapoints. We call this the *average all* anchor.

In the third variation, we deviate slightly from the standard multiview contrastive framework of [27] and use an adapting average view as our anchor. This adapting average view changes according to which view is being contrasted. To be concrete, we iterate over each view in a batch and for each view we use the average of all other views as the contrasting view for the InfoNCE loss. This differs from our second variation in that instead of a fixed full average, which will contain in it the information from any view contrasted with it, we instead remove the information from the view being contrasted. We denote this as the *average others* anchor.

For each of the three variations we train a new set of encoders from scratch all under the same set of conditions other than the three variations described above.

4.2 Anchor Experiments

The recall at 1, 5, and 10 results for image and cross-lingual retrieval for our three variation can be seen in Table 4.1. Somewhat surprisingly (or perhaps unsurprisingly), image retrieval performance increased slightly in most instances in the image anchor training scheme.

In one sense, it would seem natural that solely focusing on the image retrieval task in the loss function should produce better image retrieval scores. However, when viewed from the lens of shared mutual information, one might expect that, at an abstract level at least, removing one language encoder's incentive to learn associations with other languages would harm the model's ability to properly encode speech into a space that is shared by all languages and images. Results from Harwath et al. [5] seemed to confirm this notion that increased cross-lingual mutual information

Image Ret	E&I.R1	H&I.R1	J&I.R1	E&I.R5	H&I.R5	J&I.R5	E&I.R10	H&I.R10	J&I.R10	Ep
Img. Anch.	**19.55%**	**16.55%**	26.30%	**44.00%**	**37.85%**	58.90%	**57.65%**	**47.10%**	**72.70%**	16
Avg.All	0.10%	0.00%	0.05%	0.45%	0.40%	0.35%	0.80%	0.60%	0.75%	37
Avg.Oth.	13.05%	12.30%	25.20%	31.80%	25.90%	55.55%	44.55%	34.60%	67.95%	23
Full-Graph	18.05%	14.70%	**28.00%**	41.55%	36.10%	**59.20%**	54.55%	45.85%	72.35%	26
Cross-Ling	H&E.R1	J&E.R1	J&H.R1	H&E.R5	J&E.R5	J&H.R5	H&E.R10	J&E.R10	J&H.R10	Ep
Img. Anch.	0.95%	1.40%	1.15%	4.40%	4.55%	4.05%	6.85%	6.70%	6.25%	16
Avg.All	0.05%	0.15%	0.20%	0.35%	0.60%	0.45%	0.70%	1.80%	1.00%	37
Avg.Oth.	8.45%	8.65%	7.35%	21.70%	25.15%	19.25%	30.75%	35.35%	27.80%	23
Full-Graph	**11.55%**	**11.40%**	**10.50%**	**27.25%**	**29.00%**	**27.30%**	**37.55%**	**41.30%**	**36.80%**	26

Table 4.1: Loss Complexity Reduction Strategies

improves image retrieval. Their experiments showed moderate to increases in relative recall scores for English and Hindi image retrieval (one direction, no averaging) when incorporating English-Hindi alignment into their loss.

Our results here show the opposite effect, though, with the addition of the Japanese view. This suggests that the complexities of aligning multiple languages in the same space interfere to some extent with the visual grounding objective. This point of view is somewhat supported by the cross-lingual retrieval scores of the image anchor framework. Even without the cross-lingual loss terms, one might expect that cross-lingual retrieval would not suffer too greatly since all views must be mutually close to the same point (the image vector). However, cross-lingual retrieval scores plummet without the direct objective.

Our second variation was unable to learn under its respective setting. This is perhaps due to instability caused by having the representation of a view present in both sides of the contrasting loss. Further investigation is needed to properly explain this though.

The third variation, though, managed to learn but was consistently outper-

formed by the image anchor scheme in image retrieval. Looking at the cross-lingual retrieval results in Figure 4.1 tells a slightly different story. While the image anchor scheme's performance deteriorates drastically, our third variation (the adaptive average) is able to maintain relatively strong performance as compared withe full-graph setting. This is an encouraging result as it indicates there may be potential to make improvements to the scheme (such as a more sophisticated averaging mechanism) that would further close the gap for all view pairs.

4.3 Summary

This chapter explored three avenues for reducing the complexity of the pairwise comparisons necessary for full contrastive loss. We found that using the image view as the anchor (with three language descriptions) increased nearly all image retrieval scores, contradicting previous results in the two language scenario. However, cross-lingual retrieval scores collapsed. When considering both image and cross-lingual performance, we found that contrasting each view with the average of all other views produced the best overall results, although they still lag behind that of the full-graph.

Chapter 5

Parameter Efficiency

Up to this point we have encoded each view's representation with its own dedicated encoder model. This can become quite taxing on GPU memory as the number of views to be encoded increases. One might expect though, that there may be certain sub-tasks being performed by each encoder that are shared across languages (such as recognition of common phonemes). In this chapter we explore the potential for parameter sharing among the language encoders. Previous work has also explored this topic [17], but this is the first time we are aware of that has explored this effect in visually grounded settings.

5.1 Experimental Design

Most of the experimental settings are the same as described Section 4.1, however in this chapter we only use the full-graph training scheme. We use the Adam optimizer, a learning rate of .001, the average pooled model output as each view's final representation, and the InfoNCE loss. This isolates our experimental variable: parameter sharing design. The specifics of each design are detailed below.

5.1.0.1 Experimental Variable

We experiment with three variations of a shared language encoder. In the *basic* variation, we simply have one shared audio encoder and feed all language input into it.

The second and third variations are inspired by Johnson et al. [17] who prepend a language specific token to a multilingual neural translation model. Unlike [17], we do not use recurrent network, but a CNN. More importantly, though, our encoder operates on the speech signal, which contains orders of magnitude more time steps. We therefore explored two alternate options for injecting prior language knowledge into our CNN encoder.

For both of these variations, we started by appending a learned language-specific embedding to the log Mel filter bank (LMFB) spectrogram feature dimension of the input. Since each input utterance contains two dimensions/axes (the first being the number of time steps and the second the LMFBs), this means we concatenate the language embedding to the second dimension. This required expanding the filter sizes of the first convolutional layer. We chose this strategy appending as appending language information to the input feature dimension seemed a natural choice. Appending it to the sequence dimension is another option, but we did not explore it. We chose to use a small eight dimension embedding for all the language embedding.

Next we applied additional layer-specific and language-specific embeddings to each of the intermediate representations between the four residual blocks of the encoder. However, the decision over which dimension/axis to append the layer and

language specific embedding is not obvious. Thus, the remaining two variations we explore in our experiments involve 1) appending the language embedding to the first dimension (after the batch) and 2) appending the embedding to the last dimension. We refer to these two variations as the *channel tag* and *sequence tag* respectively.

Note that our encoder's 1D convolutions after the first are implemented using 2D convolutions and kernels with a 1 in the height dimension. Hence, when we refer to the 'first' dimension this is typically called the channel dimension in conventional CNN nomenclature. Likewise, the 'second' dimension refers to what is typically called the 'width' dimension and plays the role of the current downsampled sequence length. The 'height' dimension is one for all intermediate representations.

Since the channel embedding variation changes the effective number of features used (and necessarily then increases the filter sizes of all intermediate layers), it has approximately 150 thousand more parameters than the sequence embedding variation

5.2 Shared Encoder Experiments

The recall at $K = 1, 5$, and 10 results for image and cross-lingual retrieval can be seen in Table 5.1. Most noticeably, all shared encoders struggled to compete with the fully parameterized model. Surprisingly, though, the basic shared model outperformed both the channel and sequence embedding variations. Due to time constraints we were not able to investigate this further, but we find it counter-intuitive that removing parameters and *a-priori* information about the input language would harm performance in this manner.

Image Ret	E&I.R1	H&I.R1	J&I.R1	E&I.R5	H&I.R5	J&I.R5	E&I.R10	H&I.R10	J&I.R10	Ep
Basic	11.35%	7.40%	8.60%	31.05%	21.20%	25.85%	42.70%	29.70%	35.60%	19
Seq Tag	7.35%	4.40%	4.60%	22.50%	14.10%	16.45%	32.10%	20.20%	23.20%	17
Chann Tag	2.30%	2.20%	2.25%	8.40%	8.65%	7.60%	12.80%	13.45%	12.25%	11
Baseline	**18.05%**	**14.70%**	**28.00%**	**41.55%**	**36.10%**	**59.20%**	**54.55%**	**45.85%**	**72.35%**	26
Cross-Ling	H&E.R1	J&E.R1	J&H.R1	H&E.R5	J&E.R5	J&H.R5	H&E.R10	J&E.R10	J&H.R10	Ep
Basic	2.25%	2.80%	1.35%	8.35%	10.05%	5.80%	14.10%	15.45%	10.40%	19
Seq Tag	1.55%	1.75%	1.50%	6.85%	6.60%	5.45%	11.00%	10.00%	8.45%	17
Chann Tag	0.20%	0.30%	0.15%	1.55%	1.50%	1.20%	3.10%	2.95%	1.65%	11
Baseline	**11.55%**	**11.40%**	**10.50%**	**27.25%**	**29.00%**	**27.30%**	**37.55%**	**41.30%**	**36.80%**	26

Table 5.1: Shared Encoder Strategies

. One encoder for all languages. Cross-lingual retrieval resutls.

Another surprising result is that the sequence embedding variation performed much better than the channel embedding variation despite having more parameters. Recall that the channel dimension is effectively the feature dimension here.

5.3 Summary

This chapters experiments show that while parameter sharing might well be possible, dedicated language encoders thoroughly outperform basic sharing mechanisms. Splitting the decoder into shared layers and dedicated layers is a potential next step for exploration, as is something akin to adapter layers as proposed by Kannan et al [18].

Chapter 6

Conclusion

This thesis explored three areas related grounded multi-lingual image and utterance retrieval: loss function optimization, loss complexity, and parameter sharing. In the first we found that Masked Margin Softmax losses with variable margins consistently achieved the best retrieval scores. We also identified potential areas of improvement for these two losses. Additionally, we found that reducing the loss complexity and using an image view anchor improved image retrieval results, contrary to previous results in this area in the bilingual setting. We also presented an adaptive anchor scheme that minimized performance losses when compared with full-graph optimization. Lastly, our shared encoder experiments indicate that dedicated language encoders do, in fact, appear necessary for optimal alignment, at least compared with basic encoder sharing strategies.

Potential future extensions of this work include further exploration of parameter sharing. One interesting idea is to experiment with larger shared language encoder model. Since sharing an encoder for K languages reduces the number of total model parameters by a factor of K, one can increase the size of the encoder while still achieving a net reduction in parameters. Another direction would be to analyze the behavior of the share encoder to understand why the best performing language using

distinct encoders was suddenly the worst performing using a shared encoder.

Additionally, the results from Chapter 4 suggest that adapter heads for each view-pair may be useful during joint (full-graph) optimization. This introduces interesting possibilities for visual grounding of cross-lingual retrieval since there would now be two different feasible points in the language encoder pipeline in which to include the loss w.r.t. the visual modality. Of course, once this area is fully explored, it would also be exciting to see the impact of the using the results to optimally align intermediate representations of encoder-decoder models in MT with visually grounded context.

Bibliography

[1] Ozan Caglayan, Pranava Madhyastha, Lucia Specia, and Loïc Barrault. Probing the need for visual context in multimodal machine translation. *arXiv preprint arXiv:1903.08678*, 2019.

[2] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[3] Raj Dabre, Chenhui Chu, and Anoop Kunchukuttan. A comprehensive survey of multilingual neural machine translation. *arXiv preprint arXiv:2001.01115*, 2020.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[5] David Harwath, Galen Chuang, and James Glass. Vision as an interlingua: Learning multilingual semantic embeddings of untranscribed speech. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4969–4973. IEEE, 2018.

[6] David Harwath and James Glass. Deep multimodal semantic embeddings for speech and images. *2015 IEEE Workshop on Automatic Speech Recognition and*

Understanding (ASRU), pages 237–244, 2015.

[7] David Harwath and James Glass. Towards visually grounded sub-word speech unit discovery. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3017–3021. IEEE, 2019.

[8] David Harwath, Wei-Ning Hsu, and James Glass. Learning hierarchical discrete linguistic units from visually-grounded speech. *arXiv preprint arXiv:1911.09602*, 2019.

[9] David Harwath, Adria Recasens, Dídac Surís, Galen Chuang, Antonio Torralba, and James Glass. Jointly discovering visual objects and spoken words from raw sensory input. In *Proceedings of the European conference on computer vision (ECCV)*, pages 649–665, 2018.

[10] David Harwath, Antonio Torralba, and James R Glass. Unsupervised learning of spoken language with visual context. *Neural Information Processing Systems Foundation, Inc.*, 2017.

[11] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020.

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

[13] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018.

[14] Po-Yao Huang, Junjie Hu, Xiaojun Chang, and Alexander Hauptmann. Unsupervised multimodal neural machine translation with pseudo visual pivoting. *arXiv preprint arXiv:2005.03119*, 2020.

[15] Gabriel Ilharco, Yuan Zhang, and Jason Baldridge. Large-scale representation learning from visually grounded untranscribed speech. *arXiv preprint arXiv:1909.08782*, 2019.

[16] Aren Jansen, Manoj Plakal, Ratheet Pandya, Daniel PW Ellis, Shawn Hershey, Jiayang Liu, R Channing Moore, and Rif A Saurous. Unsupervised learning of semantic audio representations. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 126–130. IEEE, 2018.

[17] Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351, 2017.

[18] Anjuli Kannan, Arindrima Datta, Tara N Sainath, Eugene Weinstein, Bhuvana Ramabhadran, Yonghui Wu, Ankur Bapna, Zhifeng Chen, and Seungji Lee.

Large-scale multilingual speech recognition with a streaming end-to-end model. *arXiv preprint arXiv:1909.05330*, 2019.

[19] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137, 2015.

[20] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[21] Mathew Monfort, SouYoung Jin, Alexander Liu, David Harwath, Rogerio Feris, James Glass, and Aude Oliva. Spoken moments: Learning joint audio-visual representations from video descriptions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14871–14881, 2021.

[22] Yasunori Ohishi, Akisato Kimura, Takahito Kawanishi, Kunio Kashino, David Harwath, and James Glass. Trilingual semantic embeddings of visually grounded speech with self-attention mechanisms. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4352–4356. IEEE, 2020.

[23] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[24] Matthew Schultz and Thorsten Joachims. Learning a distance metric from relative comparisons. *Advances in neural information processing systems*, 16:41–48, 2004.

[25] Richard Socher, Andrej Karpathy, Quoc V Le, Christopher D Manning, and Andrew Y Ng. Grounded compositional semantics for finding and describing images with sentences. *Transactions of the Association for Computational Linguistics*, 2:207–218, 2014.

[26] Tejas Srinivasan, Ramon Sanabria, Florian Metze, and Desmond Elliott. Fine-grained grounding for multimodal speech recognition. *arXiv preprint arXiv:2010.02384*, 2020.

[27] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pages 776–794. Springer, 2020.

[28] Michael Tschannen, Josip Djolonga, Paul K Rubenstein, Sylvain Gelly, and Mario Lucic. On mutual information maximization for representation learning. *arXiv preprint arXiv:1907.13625*, 2019.

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[30] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pages 9929–9939. PMLR, 2020.

[31] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. *Neural*

Information Processing Systems Foundation, 2014.