# POSE-AWARE EMBEDDING NETWORKS
# AND
# MULTI-MODAL IMAGE-LANGUAGE RETRIEVAL

by

Domenico Curro
Honours B. Sc., Computer Science, Ryerson University, Canada, 2014

A thesis
presented to Ryerson University
in partial fulfillment of the
requirements for the degree of
Master of Science
in the program of
Computer Science

Toronto, Ontario, Canada, 2019

## AUTHOR'S DECLARATION FOR ELECTRONIC SUBMISSION OF A THESIS

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I authorize Ryerson University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I further authorize Ryerson University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

I understand that my thesis may be made electronically available to the public.

POSE-AWARE EMBEDDING NETWORKS

AND

MULTI-MODAL IMAGE-LANGUAGE RETRIEVAL

Master of Science in Computer Science, 2019

Domenico Curro

Ryerson University

## Abstract

Inspired by recent work in human pose metric learning this thesis explores a family of pose-aware embedding networks designed for the purpose of image similarity retrieval. Circumventing the need for direct human joint localization, a series of CNN embedding networks are trained to respect a variety of Euclidean and language-primitive metric spaces. Querying with imagery alone presents certain limitations and thus this thesis proposes a multi-modal image-language embedding space, extending the current model to allow for language-primitive queries. This additional language mode provides the benefit of improving retrieval quality by 3% to 14% under the hit@k metric. Finally, two approaches are constructed to address the issues of conducting partial language-primitive queries, with the former generating maximally likely descriptors and the latter exploiting the network's tendency to factorize the embedding space into (mostly) linearly separable sub-spaces. These two approaches improve upon recall by 13% and 17% over the provided baselines.

## Acknowledgements

Thank you to my advisor, Dr. Konstantinos G. Derpanis, for providing me the resources needed to achieve my goals. Thank you to my parents, Gina and Fred Curro, for providing me with a place to go whenever I needed comfort and piece of mind. Thank you Xiaochen Yuan, for being attentive and willing to listen to my problems, and always providing me with great support, companionship, and advice (all of which I should have taken sooner). Finally, thank you, and my sincerest apologies to my Nonna for not visiting you as often as I should have; graduate school was all encompassing, but I will find a way to make it up to you.

# Table of Contents

# List of Tables

# List of Figures

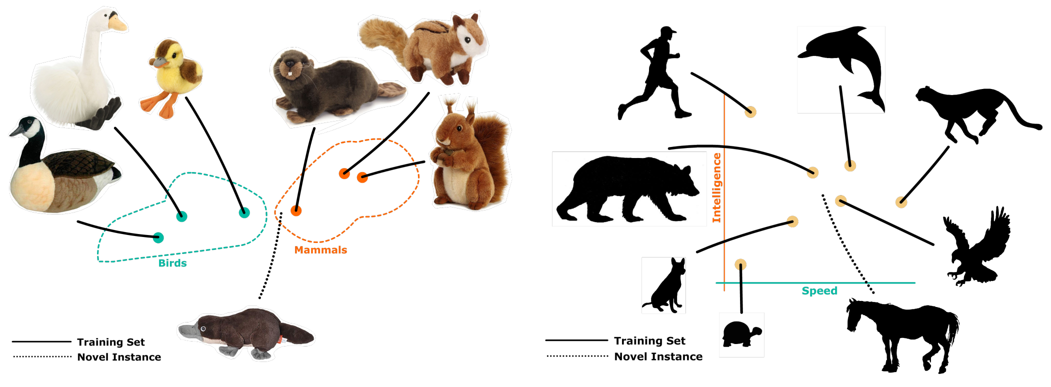
# Chapter 1

# Introduction

From web searches of recipes, to restaurant recommendations, ranking, sorting, and retrieving relevant information has become a cornerstone of digital society. This thesis conducts an exploration of fine-grained human pose retrieval across a variety of metrics, and provides a mechanism for retrieving relevant entries with both image, and language-primitive queries.

The most common approach to understanding human pose involves locating limbs or joints. While largely successful, direct joint localization from image data may fail due to occlusion or low visual quality. More so, even with correctly located joints, this framing does not immediately lend itself to the desired retrieval task. Alternatively, this thesis follows the approach outlined in [44] and [62], which learns an embedding space that respects an underlying pose-similarity metric.

An embedding space is a high-dimensional Euclidean space which approximates a desired metric by preserving its semantic ordering. This approach maps data to vectors of real numbers (each of which is called an embedding vector), with the distance between any pair of vectors implying relative sim-



(a)

(b)

**Figure 1.1:** A depiction of two possible similarity embedding spaces defined over discrete and continuous metrics. Left: An embedding space of which similarity is defined solely by class identity. Depicted here is the two class categories, birds and mammals. While the platypus is certainly a mammal, it shares visual similarity with both a duck and a beaver, and thus will locate somewhere in between both entries. Right: An embedding space of which similarity is defined by a combination of continuous factors, including speed and intelligence.

ilarity as defined by the metric. The usefulness of embedding spaces stems from their co-location of related entries, reducing similarity retrieval to a simple Nearest Neighbours search. Figure 1.1 provides two examples of possible embeddings spaces, with the first being defined by similarity in class identity, and the second being defined by similarity over a variety of factors including intelligence and speed. Each example embedding space provides a mechanism for either predicting the class, or providing a loose understanding of the traits of unseen instances.

The goal of most models is to correctly disentangle input data into a useful and concise representation. For example, the express goal of action recognition is to transform an entire video volume into a one-hot representation over the entire set of possible action classes; object boundary detection renders a binary decision as to whether a pixel marks the separation between two distinct surfaces; and optical flow generation regresses a per-pixel motion vector for the purpose of understanding scene dynamics. Similarly, embedding networks condense the provided input into a concise and meaningful vector. However, unlike the extant nature of action recognition, object boundary detection, or optical flow regression representations, the meaning of any one embedding vector exists solely in relation to its neighbours.

Building on top of [44] and [62], this thesis presents a family of pose-aware embedding spaces, across a variety of Euclidean and language-primitive metrics, for images of people in a variety of poses using convolutional neural networks (CNNs).

While human joint localization provides extremely accurate positioning of an actor's pose, describing pose in terms of pixel coordinates may be overly pedantic, especially for the purpose of similarity retrieval. For example, one is likely far more interested in retrieving images of an actor whose hands are above her head, rather than the exact spatial location or relative head-hand distance. To this end, this thesis proposes a multi-modal image-language embedding space, which accommodates human friendly language-primitive pose descriptor queries, for the retrieval of semantically similar images.

The set of language-primitives of each human pose is defined by a Posebyte [72]. A Posebyte is a set of binary values (called Posebits) each of which maps to a natural language Boolean statement. Posebits are defined in one of three ways: a joint is bent beyond a threshold angle, a pair of joints are closer than a threshold distance, or a joint is further from the torso than another joint. Example Posebit angle, distance, and relative-distance natural language statements include "Left knee is bent," "Left ankle is not near right ankle," and

"Right wrist is above head." The amalgamation of Posebits into a Posebyte provides a natural language descriptor which captures the loose geometric configuration of a pose.

Much like human pose similarity over a set of joint locations in Euclidean space, similarity over the set of language-primitives is defined holistically using the Hamming distance. Given that these metric spaces are defined by a similarity over the entirety of the pose descriptors, queries are limited to fully defined pose-descriptor sets, i.e., the entirety of the pose must be defined.

Having to define the state of each language-primitive is cumbersome and usually undesirable. One is more likely interested in querying with a subset of conditions, such as "right hand is above head; left hand is above head," with little concern for the remainder of the pose. To address this concern, this thesis proposes two approaches for querying with a subset of language-primitives: Conditional Posebytes, and Query-Aware Masks. Conditional Posebytes, inspired by [82], are high quality artificial queries generated using maximum likelihood to complete a Posebyte given a subset of Posebits. Alternatively, Query-Aware Masks, inspired by [88], takes advantage of the embedding network's tendency to linearly factorize the embedding space, and thus alters the embedding space by collapsing the relevant subspace to the origin to provide a high quality location from which to start a Nearest Neighbours search.

Inspired by the success of neural networks across a variety of tasks, this work builds on top of the convolutional neural network [48, 43, 11, 80, 83, 34]. The neural network is a feed forward model, which progressively transforms the input data into a form which is useful for some final task, and is trained using the backpropagation algorithm [74]. The convolutional neural network is a neural network designed specifically for image processing, swapping out the fully-connected layers with convolutional layers.

Each layer of the network transforms its input into a more "useful" intermediate representation, such that each subsequent representation should ease the challenge of addressing the final goal of the network. Deep neural networks are termed "deep" because they are composed of a hierarchy of layers. As the network grows deeper, as to does the richness of each subsequent intermediate representation. Each layer has the opportunity to learn more abstract concepts [5, 105, 77, 28, 39, 19, 56, 78, 65, 2] with the earlier layers capturing edges, and corners [43, 102, 24, 81], and the later layers learning shapes, parts, and objects [55, 102, 27, 103].

Taking advantage of modern advancements in neural network architectures and metric learning, this thesis explores the retrieval of semantically similar human pose images, over a variety of Euclidean and language-primitive metrics, and provides a means of querying using complete or partial language-primitive pose descriptors.

## 1.1 Motivation

For the purpose of motivating the use of pose-aware embedding networks for human pose retrieval tasks, consider a common alternative approach: joint localization.

Pose similarity, between any two images of people in a dataset, can be determined by first regressing the position of a predefined subset of rough skeletal joint locations, followed by a post-processing similarity computation composed of calculating the average distance between the respective joint locations, across the combination of each dataset entry pair. While this approach will likely be effective, a pose-aware embedding space, defined under the same metric, circumvents the need for both the direct joint localization and post-processing similarity computation steps.

Further, the minutiae of near-perfect joint localization isn't necessarily preferable. Having the knowledge that an actor's right hand is 0.1 meters above her head does not necessarily contain more meaning than the same actor's right hand being 0.2 meters above their head. In the case of natural language, the only point of interest is likely that the actor can be described as having her right hand above her head.

## 1.2 Contributions

This thesis provides the following contributions:

1. A family of pose-aware embedding networks which approximate a variety of human pose similarity metric spaces. These embedding networks provide a way to co-locate semantically similar images, for the purpose of retrieval.
2. A common pose-aware image-language multi-modal embedding space which serves as a tool for the retrieval of semantically similar images, for both image or language-primitive queries.
3. Two approaches for circumventing the limitations of holistically defined pose-aware image-language embedding spaces, for the purpose of querying with language-primitive subsets.
4. A dataset of approximately 24,000 standardized human pose images with 2D and 3D joint locations, as well as a set of language-primitive Posebyte descriptors [72], used to facilitate the training of pose-aware embedding networks.

## 1.3 Outline of Thesis

This thesis is organized as follows:

- Chapter 2 provides background knowledge for neural networks, convolutional neural networks, and embedding spaces.
- Chapter 3 presents a literature review, providing a comparison of the work conducted in this thesis with existing metric learning, pose descriptors, and multi-modal image-language research.
- Chapter 4 outlines how pose-aware embedding network architectures are defined and implemented.
- Chapter 5 introduces a pose similarity dataset, and presents a quantitative and qualitative empirical evaluation into the capacity of embedding networks to approximate the semantic ordering of a variety of metric spaces. Further, this chapter explores the two proposed approaches for circumventing the limitations of holistically defined embedding spaces for the retrieval of images with queries composed of language-primitive subsets.
- Chapter 6 provides a summary of the proposed contributions and discusses potential future paths for this work.

# Chapter 2

# Background

The following subsection provides an overview of learning in the context of neural networks, a summary of Convolutional Neural Networks, and an explanation of embedding spaces.

## 2.1 The Neural Network and Learning

The neural network is a feed forward algorithm composed of layers of small modules (neurons), which progressively manipulate the output of the previous layer and forward it to the following layer. The origin of the neural network began in the late 1950s with the invention of the Perceptron [73], receiving renewed interest [97, 46, 74] in the late 1980s due to the rediscovery of the backpropagation algorithm [75], and finally rose to prominence with a demonstration of efficacy [43] in 2012.

Neural networks are composed of small modules named neurons, a biologically inspired unit which loosely imitates the neurons of the human brain.

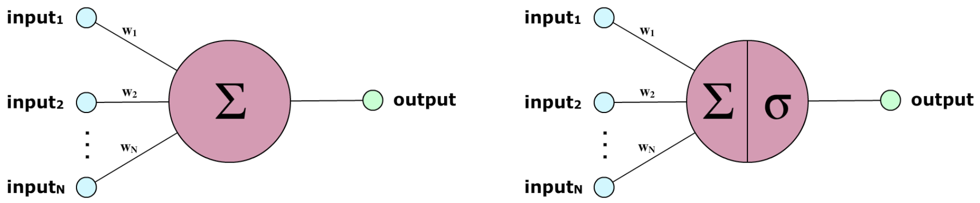


**Figure 2.1:** Left: An abstract representation of a linear neuron. The linear neuron performs a weighted summation $\sum$ of the $input_n$ values with weights $w_n$, to produce activation *output*. Right: a depiction of a non-linear neuron, where the weighted summation $\sum$ is followed by a non-linear function $\sigma$.

Neurons perform a weighted summation of the input data,

$$o = \sum w_i d_i + b, \tag{2.1}$$

where $w_i$ and $d_i$ is the neuron's $i^{th}$ weight and corresponding $i^{th}$ input feature, and $b$ is the bias. The output of a neuron $o$ is called an activation, termed so because a neuron produces a larger output for data which strongly correlates with its weights. More commonly, neurons applying a non-linear function, indicated here as $\sigma$,

$$o = \sigma(\sum w_i d_i + b). \tag{2.2}$$

Figure 2.1 illustrates two neurons, with the former being a simple linear neuron, and the latter being non-linear.

Neural networks combine neurons into layers, with each layer processing the previous layer's output. Figure 2.2 illustrates a simple two layer network which processes its input, forwarding two activations to the second layer, which in turn process its input to generate the final activation. Consecutive linear operations can be restated as a single linear transformation and thus multi-layer networks are generally composed of non-linear neurons to allow for

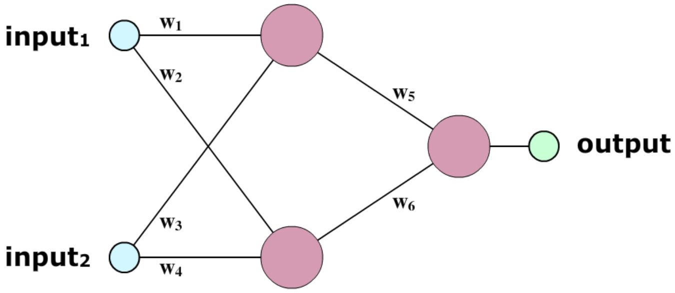


**Figure 2.2:** A depiction of a simple two-layer network. The first layer accepts a two-feature input, forwarding two activations to the second layer. The second layer takes the non-linear combination of its inputs to produce the final activation.

greater model expressiveness and the capacity to handle non-linearly separable problems.

Single layer networks are limited in their capacity to separate non-linear data. Consider the example presented in the top row of Figure 2.3. The left image depicts the non-linear two-class input data, with the positive class ✕ residing in-between the bi-modal negative class ●. Using the network presented in the right image, two possible weight configurations and their activations are plotted in the middle row. Neither model is successful in separating the data according to the desired threshold. However, a simple two layer model, such as the one presented in the bottom row, can take the non-linear combination of both proposed models to correctly separate each class.

Until the rediscovery and application of the backpropagation algorithm for neural networks, it remained unclear how to efficiently select neural network parameters (weights). Backpropagation [46, 74] is the process of efficiently calculating the partial derivative error gradient of each neuron, backwards through the network, using the chain rule.

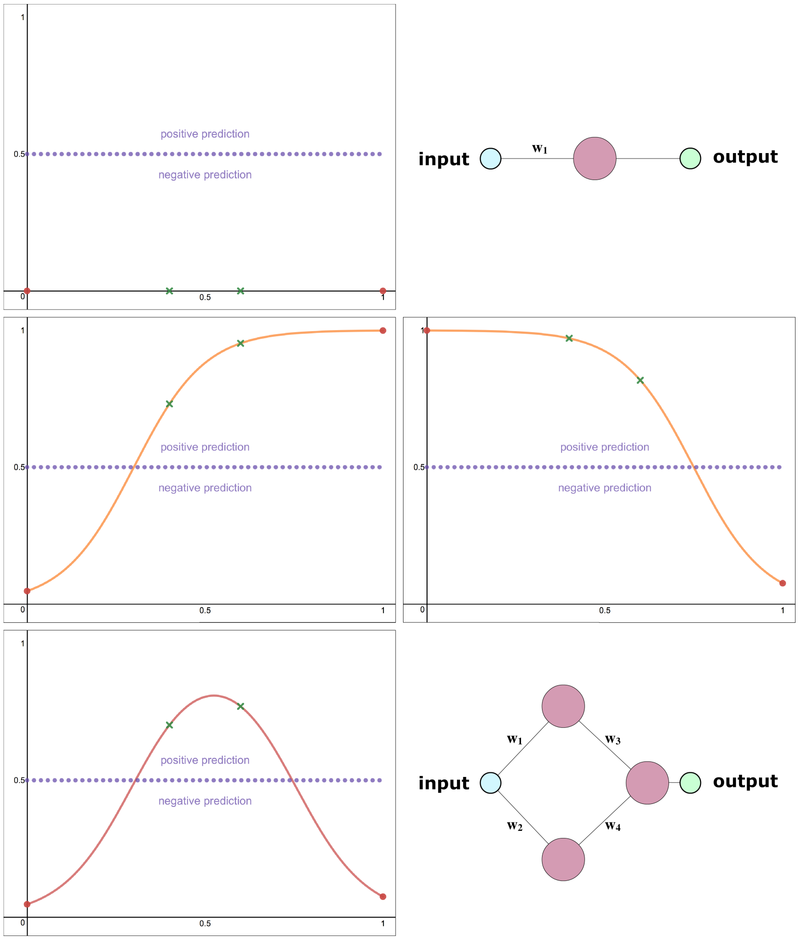


**Figure 2.3:** A comparison of the capacity of single and multi-layer neural networks. Top: a set of input data, their logistic activation, the desired separation threshold, and a single layer network. The x-axis and y-axis depicts the input values and their activation outputs, respectively. This classifier predicts any input as being part of the positive ✕ or negative ● class if its activation is greater or less than 0.5, respectively. Middle: two possible weight configurations of the single layer network. Neither model is capable of correctly separating the multi-modal input data due to the monotonic nature of the logistic activation function. Bottom: using the weight configurations from the previous two networks as the first layer and computing a non-linear combination with the second, the presented two layer network is able to correctly separate the positive and negative classes.

Training a neural network involves three steps: the forward pass, a per-weight error gradient calculation, and finally a weight update.

Figure 2.2 illustrates a simple two layer network, which results in a single output given two inputs. On the forward pass, the input data is sequentially transformed by each neuron, until the final output is produced. The per-weight error gradient is calculated using the chain rule. For example, the final output error gradient of $w_1$ is calculated as,

$$\frac{\delta E}{\delta w_1} = \frac{\delta E}{\delta o_2} \frac{\delta o_2}{\delta w_5} \frac{\delta w_5}{\delta o_{1,1}} \frac{\delta o_{1,1}}{\delta w_1}, \tag{2.3}$$

where $E$ is the model error given some input, $w_k$ is a weight parameter, and $o_{layer,index}$ is the output of the $i^{th}$ neuron at a specific layer. The use of the backpropagation algorithm assumes that each module represents a smooth (or piece-wise smooth) function in order to calculate the error gradient.

Finally, each weight is updated. While there are commonly used alternative optimization methods [3, 84, 42, 21], for simplicity consider gradient descent,

$$w_k \leftarrow w_k - \mu \frac{\delta E}{\delta w_k}, \tag{2.4}$$

where $w_k$ is the network's $k^{th}$ weight, $\mu$ is the learning rate, and $E$ is the total network error. The hyperparameter $\mu$ scales the rate at which each weight is updated. While gradient descent is a greedy algorithm, prone to local minima of the non-convex error surface, recent work has set out to understand why the local minima are almost always of high quality [13, 66, 52, 67].

## 2.2 Convolutional Neural Networks

A fully-connected neural network layer connects each neuron to each input, ignoring the potentially helpful structure of the data. For example, images

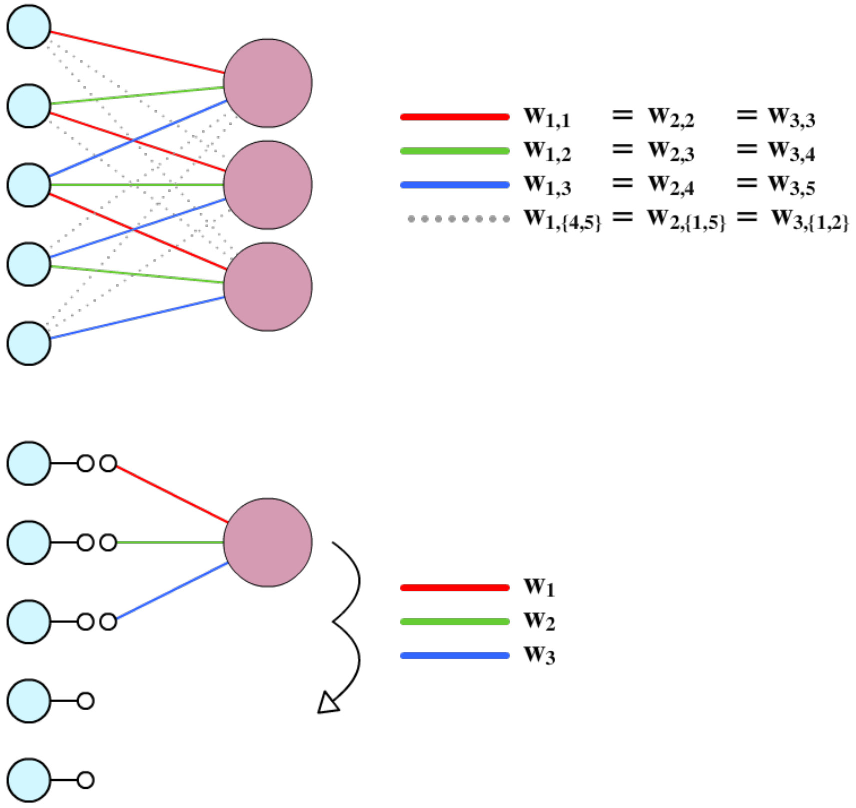

$w_{1,1} = w_{2,2} = w_{3,3}$
$w_{1,2} = w_{2,3} = w_{3,4}$
$w_{1,3} = w_{2,4} = w_{3,5}$
$w_{1,\{4,5\}} = w_{2,\{1,5\}} = w_{3,\{1,2\}}$
$w_1$
$w_2$
$w_3$

**Figure 2.4:** A depiction of convolution as a special case of a fully-connected layer. Top: A fully-connected layer, such that each neuron's weights are arranged in such a way as to apply convolution. Bottom: A 1D convolutional operation with a window size of three. Convolution runs a single neuron over the input data. While the output will be identical in both cases, the convolutional operation requires far fewer parameters (one quarter, in this particular case) when compared to a regular neural network layer. This difference in parameter count is due to the fact that regular neurons are fully-connected to their input. One obvious drawback is that, since the dotted-line weights have a value of zero, there is no value gained for a large part of the computation.

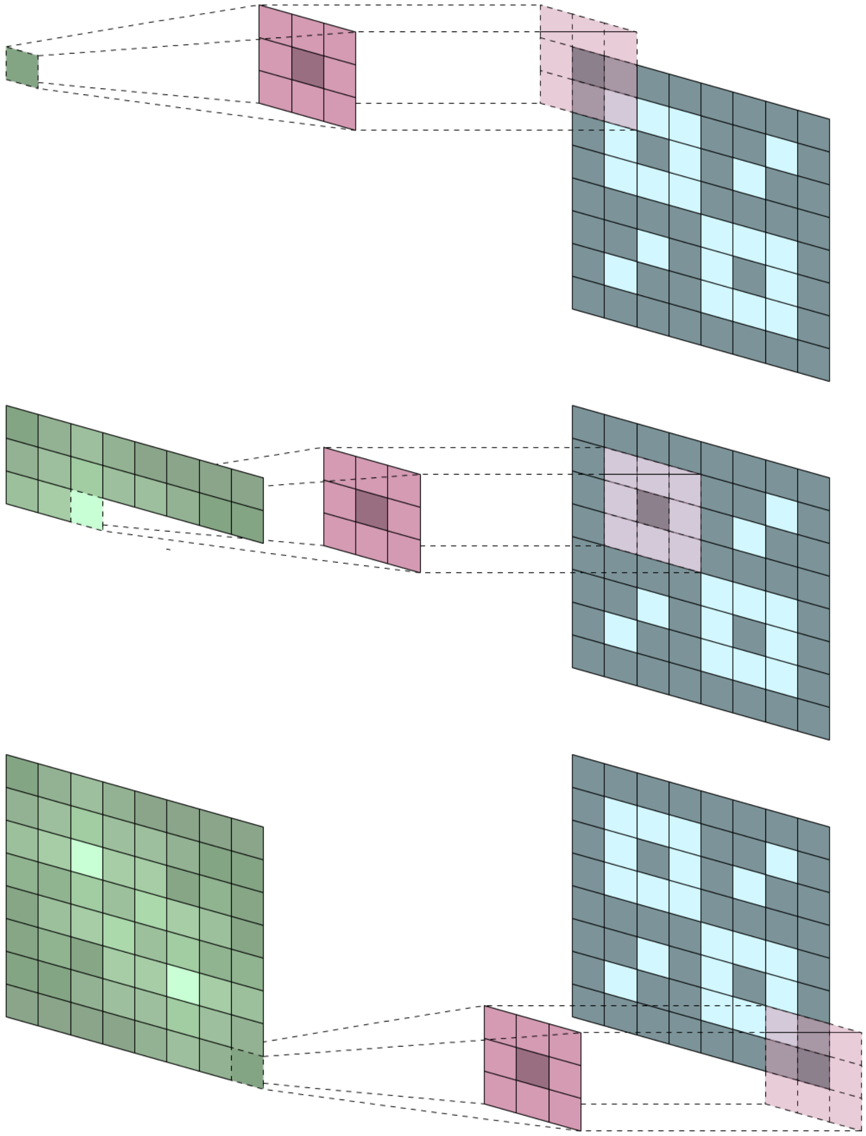


**Figure 2.5:** An example of a convolutional kernel (middle) being evaluated on a feature map (right), resulting in a new feature map (left), at three time-steps. Convolution runs a sliding window over the input feature map, producing an activation at each step. The top, middle, and bottom row depict the first, $19^{th}$, and $64^{th}$ time-step. The colour is merely to help distinguish between input, kernel and output features, however a larger intensity (brighter value) describes a stronger template match.

commonly exhibit a repetition of localized patterns, which are in general spatially invariant. Consider the case of the eye, which is a relatively small, white ovular shape with a dark round circle in the center. Being able to successfully "learn" to detect eyes may be useful for tasks, such as classification; humans have eyes, cars do not. Learning to detect an eye in a spatially invariant way is likely a difficult task for any one layer in a regular neural network. One highly common and largely successful solution is the convolutional operator.

**The convolutional layer** Convolution is the process of sliding a window over the spatial component of the input data, generating an activation at each step. Convolution extends over the spatial domain, with the discrete formulation being

$$(f * g)[n] = \sum_{i=-\infty}^{\infty} f[i]g[n-i], \tag{2.5}$$

where the filter $g$ is centered at point $(n)$ of data $f$, computing a single output as the summation of the point-wise multiplication between the two.

As depicted in Figure 2.4, a single convolution can be considered a special case of a fully-connected layer. While it is in the fully-connected layer's capacity to learn convolution, by tying the weights of each neuron in the correct configuration, it would require far more parameters, most of which will be zero, and is thus computationally more expensive and memory intensive.

The output activation structure of a convolution depends upon the induced semantic structure of the input data. For images, convolution is performed as a sliding window over the spatial extent of the data, as depicted in Figure 2.5.

Convolution of an image is similarly formulated as Equation 2.5,

$$(f * g)[n,m] = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} f[i,j]g[m-i,n-j], \tag{2.6}$$

where the filter $g$ is centered at pixel $(m,n)$ of image $f$. The final output of

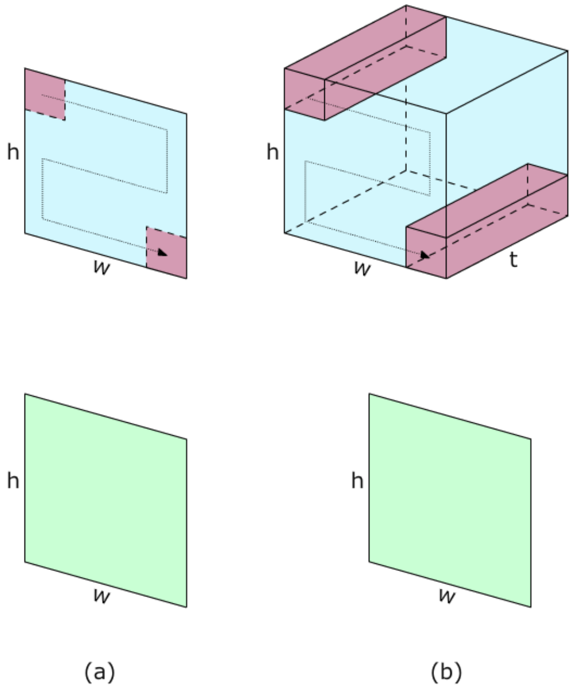


**Figure 2.6:** The convolution model (above) and its' output (below) performed on **(a)** an image, and **(b)** a concatenation of feature maps with a filter that extends across the entire feature channel extent, both of which result in a single 2D feature map.

convolution at any one pixel is calculated by taking the point-wise multiplication across the entire spatial domain, however in practice the extent considered is usually over a finite window.

For the purpose of simplicity, most depictions of image convolution merely show the spatial extent of the data. That is, convolution is illustrated as a 2D window, when in fact it is truly 3D. For example, colour images are composed of multiple (usually three) colour channel pixels, and thus to preserve the inherit cross-channel relationship, convolution generates a single activation for each discrete location in only the spatial domain, as depicted in Figure 2.6a. The possible input features to the convolutional operator are not limited to multi-channel colour images. For example, a subset of video frames may be concatenated into a volume with the network observing the sequence as a whole [20], and multiple concatenated optical flow feature maps provide a supplementary perspective of the original frame sequence [79, 93]. These alternative input formats are processed in the exact same way as image data, with the values laying under each spatial position being treated as an extended colour channel, with convolution producing a single activation at each pixel center, as depicted in Figure 2.6b.

**Convolutional neural networks** This family of neural networks replace almost every fully-connected layer with a convolutional layer. By taking advantage of the inherent structure of image data, Convolutional Neural Networks (CNNs) significantly reduce the number of model parameters, and have thus significantly outpaced conventional approaches on a variety of visual tasks, such as human pose estimation [86, 9], semantic segmentation [53], and object recognition [43, 34]. For example, on the prominent ILSVRC ImageNet [17] classification challenge, the AlexNet [43] model was able to outperform the next top contender by a striking 9.8%, with a top-5 error of 16.4%. The ImageNet classification challenge measures a model's ability to correctly predict, within five attempts, the contents of each test set image.

These networks are preferable for tasks where the desired output is image-like, such as object boundary detection [99] or semantic segmentation [53], as convolution and thus CNNs, are invariant to input size and approximately preserve the spatial structure of the data. Alternatively, CNNs may be used as a feature extractor for tasks which take images as input but require that the output be of a different shape. For example, $k$-way classification [43, 80] structures class predictions as a flat vector representation from which the final class is determined using the *argmax* operation.

While there has been steady research into model alternatives and configurations, a few notable architectures have emerged, which include: LeNet [48], an early network with two convolutional layers evaluated on a contemporaneously large dataset of small hand written digits; AlexNet [43] received praise for demonstrating the efficacy of CNNs, on the large and diverse ILSVRC [17] (ImageNet) challenge, with an architecture three layers deeper and with far more convolutional filters per layer than LeNet; VGGNet outlines a deep and highly uniform network with many spatially small 3x3 filters; GoogLeNet [83] which seeks to reduce parameter size of the model with 1x1 dimensionality reduction convolutions; and finally, ResNet [34] which addresses the vanishing gradient problem [31, 35, 70] by having subsequent blocks of convolutional layers merely learn a residual change, resulting in easier gradient flow and allowing for an enormous 152 layer deep network architecture.

## 2.3 Embedding Spaces

An embedding space is a high dimensional Euclidean space, where the distance between any pair of points alludes to their latent relationship. Embedding spaces tend to emerge naturally as intermediate implicitly learned network representations, or can be explicitly enforced as a final representation.

Neural Networks, at each layer, mutate the input data into a form which is helpful for the final upstream task. More specifically, each CNN layer learns a set of filters which disentangle the image information by smoothly partitioning the input space into visually salient intermediate representations. For example, while these intermediate representations do not directly solve some final classification task, partitioning 'round' and 'angled' things, and 'red' and 'blue' things, will certainly assist in determining whether the input image contains an apple or a Lego brick. Implicit embedding spaces are commonly used for exploring how neural networks learn [43, 55, 39], or for the purpose of similarity retrieval [58].

One downside to learning an embedding space as a second-order task, is that any desired similarity may be difficult to recover as it is buried within the high dimensionality of the intermediate representation. Alternatively, there has been a modest amount of work exploring the capacity of neural networks to approximate metric spaces [76, 88, 44, 30, 91]. A metric space is a set for which the distance, defined by a distance function (or a metric), can be defined between each possible pair. The desired approximation may be formulated as

$$\begin{aligned} &||f(a) - f(p)||_2^2 < ||f(a) - f(n)||_2^2 \\ &\forall a, p, n \ \in D \ s.t. \ \delta(a,p) < \delta(a,n), \end{aligned} \tag{2.7}$$

where $\delta$ specifies the similarity between any two examples. $a$ defines an anchor example, and $p$, and $n$ define examples which are similar and dissimilar to the anchor, respectively. For the model $f$ to correctly respect the semantic ordering of the desired metric $\delta$, the transformed positive pair $f(a)$ and $f(p)$ must be closer than the transformed negative pair $f(a)$ and $f(n)$.

Embedding space similarity metric functions are commonly defined by either a discrete or a continuous metric. Discrete metrics generally come in the form of finite annotations for tasks, such as image classification [76, 69] or zero/one shot learning [30, 36]. Figure 1.1a illustrates an example embedding space defined by hierarchical class labels. While the unseen "duck-billed platypus" provides visual cues consistent with both "duck"-like and "beaver"-like features, an ideal mapping would produce an embedding which both respects

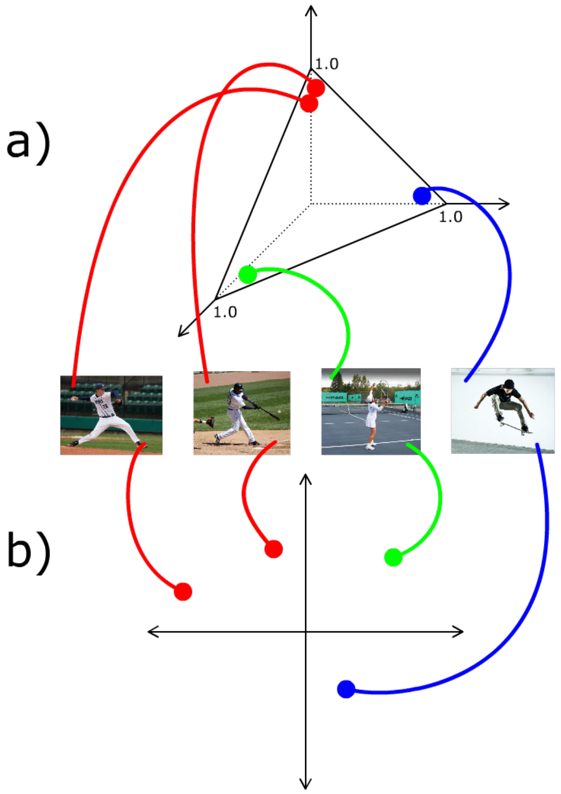


**Figure 2.7:** An example of the spaces defined by the final activation of (a) softmax for classification, and (b) a two dimensional class-similarity embedding space. Both activations express a relationship grounded in localization. Classification enforces that all points lie at the extremity of the sum-one positive hyper-plane where any straying is an indication of error. While classification designates a desired final location for each class, embedding spaces merely enforce that more similar entries lay closest to each other, with the distance between any two points alluding to some underlying similarity. In (b) both baseball players are relatively co-located, however because the baseball player swinging a bat and the tennis player swinging a racket are both holding elongated objects the distance between the two is closer than any mutually exclusive class label might suggest that they be. Instead, for (a) this similarity is completely disregarded.

these visually salient qualities and correctly co-locates it nearest to its fellow class embedding vectors. On the other hand, continuous metrics provide a relative similarity, commonly used for retrieval tasks [91, 44]. Figure 1.1b illustrates an example embedding space defined by behavioural and physiological biometrics.

The goal of learning embedding spaces is merely to preserve the semantic ordering of the defining metric, and thus the exact spatial distance between any set of embeddings does not directly correlate with their underlying features. For example, an embedding residing exactly in between two others does not necessarily mean that it is the average of its neighbours, but rather that it is merely within their upper and lower bounds.

The final embedding space representation is sometimes unbounded [30, 36], but far more often is projected onto a unit hyper-sphere [62, 69, 76, 91, 88, 89, 32]. The main benefit of unit-normalizing embedding vectors is that the similarity between any two embeddings becomes a simple angular distance.

Metric learning often takes a semi-supervised approach employing triplet rank loss [44, 76]. Unlike regression or classification for which the exact output is known, the location for any one embedding vector is arbitrary and whose meaning can only be derived from its neighbours. Classification explicitly enforces objective meaning in the output, as the final softmax activation vector is expected to hold a one in the correct index and zeros elsewhere. Figure 2.7 illustrates the geometric difference between the spaces defined by (a) the softmax activation for three-way classification and (b) a similarity embedding space. While both the batter and the tennis player are of different sport classes, they are both swinging elongated objects with the intention of hitting a ball. In the case of three-way classification, the network is expected to ignore this semantic similarity and instead enforce the maximum geometric distance between these two images, where a network trained to respect visual similarity is permitted to accommodate unanticipated cross-class commonalities.

One major benefit of embedding spaces is that they allow for the re-framing of class prediction, estimation, and similarity retrieval as a simple Nearest Neighbours search.

# Chapter 3

# Related Work

This subsection focuses on three research areas closely related to the work presented in this thesis: Metric Learning, Pose Descriptors, and Multi-modal Learning.

## 3.1 Metric Learning

Metric learning is the process of training a model to respect some desired underlying data similarity. The model learns a mapping from raw input data to an approximated metric space, of which the co-location between every embedding vector pair reflects an equivalent ordering under the desired metric. Metric learning is commonly used for tasks, such as classification, zero-shot learning, and similarity retrieval by ranking. In contrast to traditional classification or regression approaches, metric learning is considered semi-supervised as only the similarity between data points is provided rather than a concrete category label or a desired output value. While traditional approaches enforce a desired final spatial representation, such as the one-hot classification approach which enforces that each data point map to its expected corner in the hyper-plane of sum-one, learned metric spaces are free to arbitrarily locate data if it is decided that there exists some useful inter-class similarity.

While a variety of metrics have been explored, they usually come in one of two varieties: similarity by discrete labels, and similarity by continuous values.

**Discrete classes and categories** With the plethora of classification and category labeled datasets, a variety of approaches have used embedding spaces to take advantage of cross-class similarity to address problems like zero-shot learning, face recognition, and class prediction. Under a discrete metric, a pair of entries is either identically similar or not.

Embedding spaces learned under a person-identity similarity metric, using CNNs have been shown to be effective for face recognition [76, 69]. Existing vocabulary embedding spaces [59, 60] have been used to map each image closest to its prototype (class category) [30], for unseen class prediction. Similarity under the WordNet [61] defined ontological ordering, has been explored for both images and image-captions simultaneously [89]. Each of the previously described metrics are defined on a single condition: the face belonging to the same person, the image mapping closest to its prototype, and the image or caption being correctly ontologically ordered. In contrast, the pose-aware embedding networks, proposed in this thesis, are defined by a family of continuous Euclidean and language-primitive similarity metrics.

As opposed to approximating a single metric, Conditional Similarity Networks [88] simultaneously enforces a variety of mutually exclusive discrete class category metrics over the same data. For example, the relationship of the entries in a font dataset can be considered over letter identity or typeface. It demonstrates that embedding networks have the capacity to learn how to factorize concepts shared by multiple goals, by allowing the network to learn a set of per-concept partitioning masks. Allowing the network to learn and partition multiple concepts simultaneously, the final embedding space can be transformed by any one of the partitioning masks to allow for the rearrangement of entries, with a simple point-wise multiplication. The Query-Aware Mask model, proposed in this thesis, was largely inspired by Conditional Similarity Networks and thus follows a similar process of learning a set of masks to alter the final embedding space. However, where [88] may be inducing a factorization by enforcing that the model simultaneously respect multiple metrics, Query-Aware Masks demonstrate that pose-aware embedding networks exhibit a similar factorization, while being trained under a single continuous language-primitive Hamming distance metric, as demonstrated in (Section 6.2.2). Further, Query-Aware Masks demonstrate that these models have an inherent tendency to factorize concepts at a much more granular level, than demonstrated in [88], such that language-primitives appear to reside on their own linear subspace within the final embedding space representation.

**Differences in continuity** Continuous similarity describes the degree to which a pair of entities are related, usually defined over a continuous value.

Labeling images with fine-grained annotations is a tedious task and thus the vast majority of image datasets often contain a single concept describing the entire image. For example, the large scale ImageNet [17] dataset provides a single object category label per image. While discrete datasets have been used to construct retrieval systems, there is no guarantee that the network will truly learn cross-class fine-grained similarity. This concern has been addressed by producing a "golden feature" [91] [12], which is a set of off-the-shelf image descriptor generation algorithms, such as Scale-Invariant Feature Transform (SIFT) [54], Histograms of Oriented Gradients (HOG) [15], and colour histograms. More specifically, they construct a dataset from many hundreds of thousands of text queries, the results of which are then evaluated for similarity using a “golden feature,” with independent queries assumed to have no relation. These two approaches differ in that [12] operates directly on the “golden feature,” where [91] trains a fine-grained similarity retrieval network from raw image data, relating entries by their “golden feature” similarity. Further, they also propose to first learn a course image similarity by pre-training on the ImageNet [17] dataset, followed by a fine-grained network fine-tuning. While using a class label may not guarantee visual similarity, the bias found within datasets is likely helpful. For example, cows are often photographed while grazing in a lush, open field, and rarely have their picture taken while loitering on rooftop patios. In contrast, this thesis strictly explores human pose similarity over a family of Euclidean and language-primitive metrics, and thus unlike similarity defined by a “golden feature,” similarity between any two images is strictly defined by human pose and is thus robust to foreground objects, background scenery and overall image colour.

Pose annotation datasets have been used to construct pose-aware retrieval models [44, 62]. The first approach [44] enforces image-pair similarity over the mean per-joint distance of any two poses. Similarly, [62] operates over the same metric, but performs a torso alignment prior to computing the metric distance, whereas [44] merely centres the two poses. This thesis can be thought of as an extension to both [44] and [62], with a wider exploration into a variety of Euclidean and language-primitive metrics.

## 3.2 Pose Descriptors

A variety of tasks benefit from (or even depend on) understanding the underlying human pose. For example, an action recognition model might perform fairly well by merely observing background cues, such as a green open field and a baseball, but will certainly become confused if said cues are present for multiple classes, such as throwing or catching a baseball. Endowing the model with the understanding of human actors would likely help in discerning which action is occurring. To aid in a model's understanding, knowledge about human pose is commonly induced in one of three ways: geometrically, implicitly, and through natural language representations.

**Geometric representations** Likely due to their intuitive nature, these representations are the most common way of inducing explicit knowledge of human pose into a model. Geometric representations of human pose usually come in the form of human joint locations in 2D pixel coordinates [86, 9, 96, 64] or 3D metric coordinates [49, 50, 106]. In contrast to explicit joint localization, this thesis instead proposes a family of pose-aware embedding networks which learn to understand pose implicitly. That is, rather than provide the network with precise coordinates of a predefined set of joints, the models are trained with triplets composed of similar and dissimilar entries where similarity is defined by a Euclidean or language-primitive metric distance.

**Implicit representations** Prior to the development of the neural network, these representations were the most common way of making ground in human pose related tasks. Implicit representations of the human pose come in two varieties: intermediate hand-crafted pose-aware features, or intermediate hand-crafted generic features tuned in such a way that the final model is pose-aware. Both varieties operate in approximately the same manner, involving an offline selection of a set of features, processing the image or video with said feature set to generate a new representation, and finally classifying the new representation using a discriminator model, such as a Support Vector Machine (SVM) [14]. The intent of generating a new representation is to provide the discriminator with a verbose descriptor of the input, which is easier to disentangle than the original imagery.

Two popular implicit pose-aware features are Poselets [8, 6] and Actemes [104]. A Poselet is a part detector that is aware of a single subsection of the human body. While geometric interpretations often consider a single part or joint, Poselets may arbitrarily span multiple joints. For example, a Poselet can simply be a head-and-right-shoulder detector. Similarly, an Acteme is a human pose video template that is also not limited to a single part or joint, and instead captures a spatiotemporal window that overlaps a person in motion. A set of Poselets (or Actemes) generate a pose-aware feature representation of the input data. The embedding networks, presented in this thesis, similarly generates a pose-aware feature representation but with a major distinction being that the Poselet and Acteme feature generator is hand-crafted and therefore the resulting feature is well understood. In contrast, the final embedding features generated by the models proposed in this thesis remain latent.

Alternatively, a model can be composed of generic features, each of which is not inherently pose-aware. Each of these models can learn to be pose-aware with a tuning towards pose-centric tasks, such as pose estimation or action recognition. Some generic feature models include [18] which generates a fixed set of spatiotemporal energy filters, [38] an early feed forward convolutional network which selects filters in a data driven approach, [22] which computes an optical flow based motion descriptor, and more contemporaneously, CNNs [79, 93, 92, 94, 44, 86, 9] which are composed of a set of generically initialized filters which are then made relevant using the backpropagation algorithm [74]. While each feature on its own does not describe human pose, together they do indeed comprise a model which can be seen as a pose-aware. Similarly, the pose-aware embedding networks, proposed in this thesis, are constructed using

CNNs and are thus generic feature models which have been made pose-aware.

**Natural language representations** A far less common approach to comprehending imagery, likely due to the difficulty in annotation collection, natural language representations describe the contents of a scene in a human-readable way. In contrast to an object or action class label, which describe the main focus or dynamics of a scene, natural language representations generally decompose the scene into its constituent parts.

Language-primitives, each of which describes a component of the human pose, have been outlined in [72], [63], and [51]. More specifically, [51] presents a set of action primitives which decompose the entire body's motion into smaller components, like "arms moving in a pendulum motion" or "torso twist". Similarly, [72] and [63] define a set of pose primitives which describe the relationship between joints, such as "left wrist is near right hip" or "right ankle is near head." This thesis proposes to use the natural language-primitives outlined in [72] for the purpose of producing an image descriptor from which a language based similarity metric may be defined, and to extend the proposed pose-aware image embedding space to accept an additional language-primitive mode, allowing for a more natural way of conducting queries.

While not describing human pose, there has been some effort towards describing the actors themselves. One approach proposes to decompose entities into a set of common inter-class concepts [95], such as "has eye," or "has leg". Similarly, [7] decomposes people into concise descriptions, such as "has hat" or "is male". Alternatively, [82] learns a mapping between language-primitives and a set of human-mesh basis vectors to generate an avatar (a computer generated character) or invert the process to provide a language-primitive description of an avatar. In contrast, pose-aware embedding spaces are strictly pose-centric and are thus invariant of articles of clothing, gender, and body shape.

Some alternative natural language representations include image captioning and action class labels. Image captioning describes imagery in a concise natural language sentence. While a class label or action category point out the main scene contents or dynamics, image captioning is far more versatile. Generated captions [40, 90, 100] usually depict abstract overarching themes within the scene, and rarely describe human pose in a nuanced way. For example, a model may produce the caption “Baseball players standing on a field” for a corresponding image. This level of caption abstraction is likely due to the desired task of general scene descriptions. Similarly, action classes [57, 101] are largely abstract as they capture a wide range of possible pose configurations for a single class. For example, in the case of “sitting,” are the actor’s legs crossed? Are her arms relaxed on an armrest or are they folded in her lap? Is she sitting upright or is she lounging in a relaxed position? For the purpose of training fine-grained pose-aware embedding networks, neither of these natural language representations provide the required fine-grained detail.

## 3.3 Multi-modal learning

The actualization of a concept lends itself to a variety of potential representations. For example, “A father and son playing catch,” can be modeled linguistically (as stated), can be depicted as a photograph, painting, sketch, or video, and can be decomposed into a set of constituent parts {‘man’, ‘child’, ‘ball’, ‘outdoors’}. Multi-modal learning attempts to understand and model various forms of the same entity into a concise and identical cross-mode representation. It is primarily used as an alternative to traditional classification approaches, zero-shot learning tasks, and image captioning tasks, and provides a mechanism for retrieving or ranking data that may take on a variety of forms.

The most common approach to forming a cross-modal representation is by learning a common embedding space between various input streams. The process generally involves enforcing that the features generated for a single entity, across all modes, map to highly similar spatial locations within the embedding space.

Producing an embedding space defined by class similarity has been shown to be beneficial for cross-modal retrieval. This is achieved by initially learning one mode, followed by fine-tuning the remaining modes while freezing the (deeper) shared layers [10]. This thesis proposes to follow the outlined approach to create an image-language embedding space, defined initially by pose similarity in imagery, followed by a mapping from a language-primitive mode to the defined embedding space.

Preexisting word co-occurrence embedding spaces [59, 60] have been proposed to solve classification and zero-shot learning challenges by mapping images to their prototype (class labels) [30, 68, 29]. Alternatively, [26, 45] propose to unify multiple modes by learning to predict hand-crafted intermediate cross-class extant descriptors, and [90, 100, 89] create a common embedding space for natural language sentences and images, using an LSTM and a CNN, respectively. For the purpose of training pose-aware image-language embedding networks, this thesis avoids the use of word co-occurrence embedding spaces as they extend beyond pose-language, allows the network to learn a rich embedding spaces as opposed to a hand crafted one, and avoids natural language sentence captioning due to the largely generic (non-pose specific) image caption datasets.

# Chapter 4

# Technical Approach

This chapter outlines the proposed pose-aware embedding and multi-modal image-language network architectures.

## 4.1 Pose-Aware Embedding Networks

This section defines the family of explored metric spaces, and explains the objective functions for learning pose-aware and multi-modal image-language embedding networks.

### 4.1.1 Pose Similarity Metric Spaces

A metric space is a set for which the distance between all elements, defined over a distance metric, is known. Metric distance functions are commonly used for the evaluation of human pose estimation [37]. For example, the quality of predicted human joint locations is assessed by the mean per-joint error (a distance measure) with respect to the ground truth.

This thesis proposes a family of pose-aware embedding spaces which respect the semantic ordering of their corresponding metric space. While the defined metrics are similar to those used for quantitative evaluations, the proposed embedding spaces instead learn to approximate these metrics as a measure of similarity, co-locating semantically similar entries and thus simplifying the retrieval process to a Nearest Neighbours search. The explored human pose similarity metric spaces include 2D, 3D, and Procrustes Euclidean distance, as well as the Hamming distance over a set of language-primitive pose descriptors, i.e., Posebits [72].

**2D pixel Euclidean distance** 2D pose similarity, between any two dataset entries, is measured as the centered mean per-joint pixel distance:

$$\delta_{2D}(J_a, J_b) = \frac{1}{N}\sum_{i=1}^{N}\sqrt{|J_a^i - J_b^i|^2}, \tag{4.1}$$

where $J_a$ and $J_b$ are pixel joint pose annotations, with the superscript $i$ indicating the joint annotation index.

Each dataset entry is derived from a spatially square per-centered image crop which has been re-sized such that the longest side is 144 pixels wide, thus each set of joint annotations is standardized such that the largest width or height is about 115 pixels. Following the motivation provided by [44] (that modern person detectors provide a person-encapsulated bounding box), the provided joints are not standardized beyond being centered within the image.

**3D Euclidean distance** 3D pose similarity between any two dataset entries is measured as the volume centred mean per-joint distance. The 3D joints are normalized such that each person is 6-feet tall, from head-to-ankle, and each limb (connected joint-pair) is re-sized such that it conforms to the relevant mean limb length of the entire dataset. This can be similarly formulated as Equation 4.1,

$$\delta_{3D}(K_a, K_b) = \frac{1}{N}\sum_{i=1}^{N}\sqrt{|K_a^i - K_b^i|^2}, \tag{4.2}$$

where $K_a$ and $K_b$ are 3D metric joint pose annotations, with the superscript $i$ indicating the joint annotation index.

**Procrustes distance** The Procrustes Transform pose similarity between any two 3D dataset entries is measured as the volume centred mean per-joint distance, after a Procrustes Transformation:

$$\delta_{procrustes}(K_a, K_b) = \sum_{i=1}^{N}\sqrt{|K_a^i - P(K_b^i; K_a^i)|^2}, \tag{4.3}$$

where $K_a$ and $K_b$ are metric joint annotations, with the superscript $i$ indicating the joint annotation index, and $P(K_b^i; K_a)^i$ indicating the Procrustes Transform applied to $K_b^i$ given $K_a^i$.

The Procrustes Transform is computed on one of the two poses, such that the transformed pose maximally aligns with its comparator. The major distinction between the Procrustes distance metric, and the 2D pixel and 3D Euclidean distance metrics is that the Procrustes distance metric is camera invariant. For example, the same person simultaneously captured by two cameras would result in a Procrustes distance of zero, where the 3D distance for any two non-identical cameras would be larger than zero.

The Procrustes Transform is composed of three transformations: a scaling, a center of mass realignment, and a rotation. Given that each actor is assumed to be 6-feet tall from head to ankle, the scaling transformation is not applied when determining the distance between any two poses.

**Posebit Hamming Distance** Posebits [72] are a set of language-primitives expressed as a vector of binary values (a Posebyte). Each bit maps to a language-primitive statement, with a value of one or zero indicating whether the condition is true or false, respectively. For example, the pose of an actor brushing her teeth may be described by a Posebyte with bits "right elbow bent," "left foot far from right foot," and "right hand above head" as $[1, 0, 0]$, as only the first of the three conditions is true. Posebyte similarity between any two poses is measured as the Hamming distance between their descriptors:

$$\delta_{hamming}(L_a, L_b) = \frac{1}{P} \sum L_a \oplus L_b, \tag{4.4}$$

where $L_a$ and $L_b$ are Posebytes, $P$ is the number of bits in a Posebyte vector, and $\oplus$ indicates an exclusive-or operation. Posebyte similarity, like Procrustes similarity, is camera invariant; "right shoulder is bent" is true independent of camera position and orientation. More so, Posebits are also invariant to certain kinds of joint articulation. For example, the Posebit that activates when "right hand above head" is invariant to how far above the head the right hand is, and only changes in the case where the condition is no longer true.

### 4.1.2 Learning Pose-Aware Embeddings

A pose-aware embedding network attempts to approximate the semantic ordering of a desired metric space:

$$\begin{gathered} ||f(a) - f(p)||_2^2 < ||f(a) - f(n)||_2^2 \\ \forall a, p, n \ \in D \ s.t. \ \delta(a_{pose}, p_{pose}) < \delta(a_{pose}, n_{pose}), \end{gathered} \tag{4.5}$$

where $\delta$ specifies the similarity between any two examples. $a$ defines an anchor example, and $p$, and $n$ define examples which are similar and dissimilar to the anchor, respectively. For the relationship to hold, the positive pair $f(a)$ and $f(p)$ must be closer than the negative pair $f(a)$ and $f(n)$.

This relationship is enforced using the triple rank hinge loss function:

$$\mathcal{L}_{triplet} = \Big[||f(a) - f(p)||_2^2 - ||f(a) - f(n)||_2^2 + m\Big]_+, \tag{4.6}$$

where the margin $m$ enforces a minimum acceptable distance.

### 4.1.3 Learning Multi-modal Pose-Aware Embeddings

The proposed pose-aware embedding networks approximate the semantic ordering of a metric, condensing an image into a concise vector representation who's meaning is merely implied by its location in relation to neighbouring embedding vectors. When embedded, each dataset entry pair should share a proximity strongly resembling that of their underlying pose-descriptors, under the defined metric. However the underlying pose descriptors are not directly accessible from the embedding space, as the learned transformation operates strictly on image data. Querying solely with imagery presents certain limitations. For example, one might desire a set of images of actors who best match the descriptor: “left foot and right foot far apart; right hand above head; left hand not above head,” but may not have an image containing this exact criteria to use as a query.

Natural language-primitives provide a clear way of defining a query, avoiding the pedantry of having to define an actor's exact joint positions or the need of having an image which roughly matches the desired criteria. To this end, this thesis explores a multi-modal image-language embedding space for the purpose of image retrieval.

Taking advantage of the previously learned pose-aware image embedding space, defined by the Hamming distance over the language-primitive Posebyte descriptors, a new mapping is learned from $e_{pose}$ to $f(e_{image})$, where $e_{image}$ and $e_{pose}$ are the image and language-primitive descriptor pair, respectively, and $f(e_{image})$ is the embedding generated by the pose-aware image embedding network.

Given that the embedding space is regularized to be a unit hyper-sphere, a new mapping, $g$, can be learned such that the cosine distance between $f(e_{image})$ and $g(e_{pose})$ is enforced to be zero by:

$$\mathcal{L}_c = 1 - f(e_{image}) \cdot g(e_{pose}). \tag{4.7}$$

Note that the parameters of $f$ are fixed, thus the model error is strictly a function of the parameters of $g$.

Equation 4.6 and Equation 4.7 provide a common mapping to a unified embedding space such that both images and language-primitives can be semantically co-located, allowing for a more natural way of retrieving images with a simple natural language query.

## 4.2 Implementation Details

This section provides details for the practicality of training pose-aware embedding networks. The following sections delve into the network architectures for both the uni-modal and multi-modal embedding networks, and provides two approaches for querying holistically defined embedding spaces with partial language-primitive queries.

### 4.2.1 Similarity Embedding Network Architecture

To realize the desired family of similarity metrics, the VGG-S CNN architecture is considered.

**CNN architecture** Following the work of [44], a family of pose-aware embedding networks are constructed using the VGG-S Network architecture [11] (as seen in Figure 4.1). This network is composed of two major components: a fully convolutional feature extractor, and a fully-connected classifier.

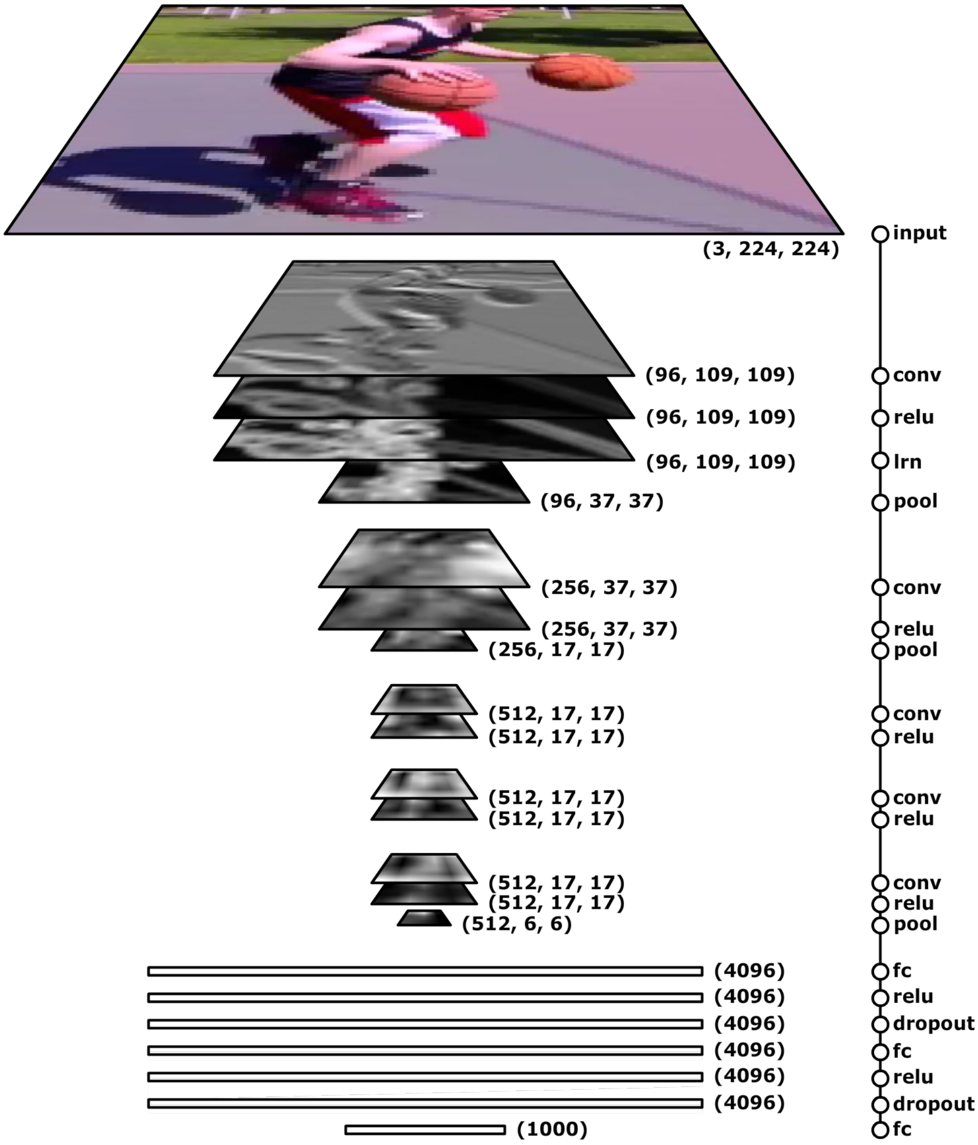


**Figure 4.1:** The VGG-S Network as described in [11].

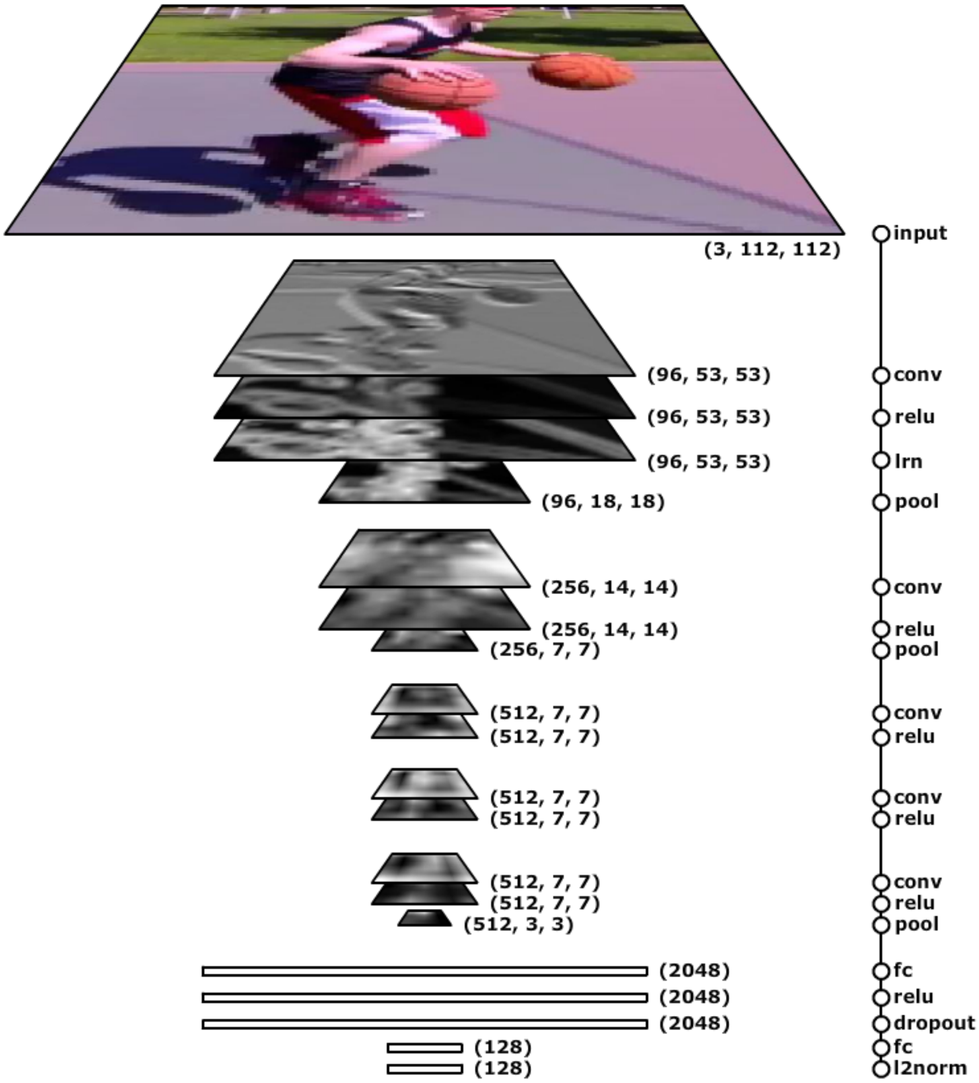


**Figure 4.2:** The pose-aware embedding network as described in [44], with the one exception being that the input size is halved to accommodate the spatial dimensionality of the dataset images. This network is a repurposed VGG-S Network [11] with the fully-connected layers having been replaced.

The former part of the network is composed of five convolutional layers. The first, second and fifth are followed by a max pooling layer which performs a down-sampling, by dividing the feature into equal window sizes and selecting the maximum activation, to produce a smaller feature map. Following the first convolution a local response normalization (LRN) layer, biologically inspired by lateral inhibition [43], normalizes cross-channel activations. This module can be interpreted as creating competition between neighbouring neurons, effectively encouraging specialization.

The latter part of the network, immediately following the convolutional component, is composed of three fully-connected layers. The first two are followed by dropout layers, which randomly suppress 50% of the neuron activations, i.e., “dropping” them. This encourages the model to use its full capacity and thereby increases generalization. The final fully-connected layer is followed by a softmax activation, providing a normalized per-class probability score.

Each convolutional and fully-connected layer in the network is followed by a ReLU activation, injecting non-linearity into the model and thus allowing the network to learn non-linear distributions.

The VGG-S Network is repurposed to create a pose-aware embedding network. The fully-connected component is supplanted with two new fully-connected layers (as seen in Figure 4.2). The new fully-connected layers are 1024 and 128 neurons wide, with the former being followed by a ReLU and Dropout component, and the latter being followed by an $l_2$ normalization such that the final activation exists on a unit hyper-sphere.

**Loss calculation** Triplet rank loss considers the relative proximity over a triplet, enforcing that the positive pair embed closer than the negative pair, within a desired margin. This loss calculation acts in a semi-supervised fashion in that no direct ground truth pose annotations are provided.

As suggested in [44], triplets are selected such that one anchor, five similar, and 105 dissimilar entries are randomly selected from the dataset. With a batch size of 111, this approach provides 525 unique triplet combinations.

The total network error is calculated across the entire mini-batch

$$\mathcal{L}_{triplet} = \frac{1}{I+J} \sum_{i=1}^{I} \sum_{j=1}^{J} \Big[ ||f(a) - f(p_i)||_2^2 - ||f(a) - f(n_j)||_2^2 + m \Big]_+, \qquad (4.8)$$

where $a$, $p$, and $n$ are the anchor, positive, and negative entries. $I$ and $J$ are the number of sampled positive and negative examples. $m$ acts as a margin ensuring that the positive pair is at least $m$ closer than the negative pair, in the embedding space.

**Curriculum learning** One possible approach for selecting a triplet, for a given anchor, could be to divide the dataset into two parts with the former being composed of entries which closely resemble the anchor, and the latter containing the remainder of the dataset. Triplets can then be composed by randomly selecting positive and negative examples from the respective subsets. This approach may be successful to some degree, but could result in early sub-optimal convergence as the network is unlikely to see challenging triplets. Consider two illustrative example anchors, with the first being a yoga pose, and the second being a camera-facing neutral stance. People are far more often found standing, sitting or walking, and thus yoga poses are uncommon. Allowing for the random selection of positive and negative examples would yield very easy triplets, posing no challenge to the network. In contrast, the camera-facing neutral stance is a very common pose. Thus, triplets composed of randomly selected positive and negative examples would not necessarily yield triplets which could inform the network of fine-grained similarity. To deal with this concern, the approach outlined in [44] proposes to use a curriculum learning scheme to provide the network with progressively more challenging triplets over the training process.

Curriculum learning [4, 23] is a regimented training scheme where the network is progressively given more challenging examples. Similar to how a student graduates through school grades, the network builds a foundation from simpler examples from which it can use to adapt to greater challenges.

This approach is realized through the selection of triplets, as outlined in [44]. For each triplet, the dataset is sorted by similarity to the anchor and divided into two parts. The former is composed of the 30 most similar dataset entries, and the latter is composed of the remainder of the dataset. To compose a triplet, a positive example and negative example is selected from their respective parts. To ease the network into the difficult challenge of learning fine-grained pose similarity from imagery alone, after every epoch (once the model has seen every image as an anchor) the most dissimilar 3,000 entries are excluded from the selection process, until there are at most 1,000 remaining. Removing these easy negative examples, over time, forces the network to consider more fine-grained pose details.

### 4.2.2 Multi-modal Embedding Network Architecture

The previously defined pose-aware embedding networks define a mechanism to co-locate similar images based on an underlying pose similarity metric, simplifying similarity retrieval to conducting a Nearest Neighbour search.

To allow for a more natural way to retrieve images, the pose-aware image embedding network trained to respect the Hamming distance similarity metric is extended to permit queries composed of a set of language-primitives.

**Image-language mapping** For the purpose of learning a mapping from

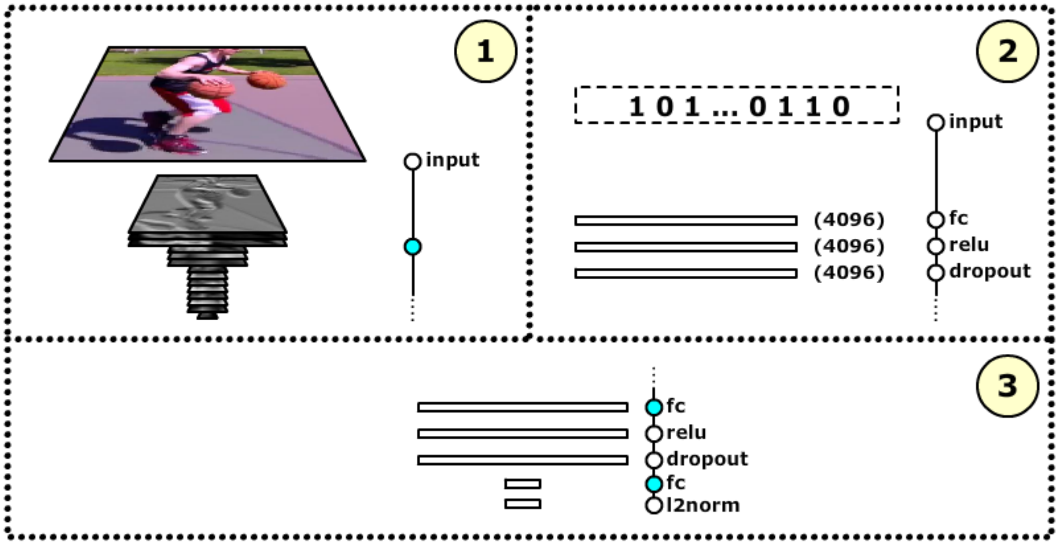


**Figure 4.3:** The proposed image-language multi-modal embedding network. 1) The network's image stream. 2) The network's language stream. 3) The common embedding space. The process of learning the proposed image-language multi-modal embedding network involves training 1+3 using triplet rank loss with similarity being defined by the Hamming distance over language-primitives, freezing the weights of both 1 and 3 (indicated in blue), and finally training 2+3 using cosine-similarity between the corresponding image and language-primitive embedding vectors.

the Posebyte [72] language-primitives to their corresponding image embeddings, a new language-primitive network is constructed to respect the cosine similarity, as state in Equation 4.7, between the embeddings generated for the two modes of the same dataset entry. The network architecture is outlined in Figure 4.3. The weights of the latter two fully-connected layers are borrowed from the pre-trained Hamming distance pose-aware image embedding network, such that the final desired representation exists within the same space as their image equivalent. To ensure a consistent mapping between both input modes, only the language stream's first layer's weights are free to change.

**Loss calculation** To learn the desired image-language mapping, Equation 4.7 is computed across each mini-batch:

$$\mathcal{L}_c = \frac{1}{N} \sum_{i=1}^{N} 1 - f(e_{image}) \cdot g(e_{pose}), \tag{4.9}$$

where $N$ is the size of the batch, $f(e_{image})$ is the image embedding vector generated by the image stream, and $g(e_{pose})$ is the corresponding language-primitive embedding vector generated by the language stream.

### 4.2.3 Language Subset Queries

The proposed multi-modal language-image embedding network provides a mechanism to conduct both image and language-primitive queries within a common embedding space. Any one location, on the unit hyper-sphere embedding space, alludes to a specific pose, with its neighbouring locations alluding to highly similar poses. This is a natural consequence of defining the similarity metric over the entire pose, i.e., two poses are deemed similar, and thus co-locate, only if a relative majority of their language-primitives agree.

Querying with a Posebyte describing an actor who is standing in a neutral position with their left elbow bent would, by definition, retrieve images of other actors standing is similar neutral stances, with their left elbows bent. This presents a limitation in that one may not be interested in the configuration of the entire pose, but instead may want to retrieve images of actors in a variety of poses but whose left elbows are bent. That is, one may wish to query with a subset of language-primitives regardless of how the remaining descriptor is defined.

Presented here are two distinct solutions for circumventing the limitations of holistically defined embedding spaces: Conditional Posebytes and Query-

Aware Masks.

**Conditional Posebytes** The proposed multi-modal language-image embedding network provides a mechanism for conducting language queries using a set of bits (a Posebyte) which are either a one or a zero, describing whether a pose based condition is true or false. One consequence of this approach is that the entire Posebyte must be defined prior to embedding. To address this problem, proposed here is a maximum likelihood approach to complete a Posebyte when given a subset of desired Posebits.

While each bit is either a one or zero, for mathematical convenience Posebytes are assumed to be defined by a Normal Distribution [82]. This assumption allows for a simple way of conditioning a Posebyte on a set of known Posebits.

The Normal Distribution $N_p$ is defined by the mean Posebyte $\mu$, and covariance matrix $\Sigma$,

$$\Sigma = P'^T P', \tag{4.10}$$

where $P'$ is defined as the mean centred Posebyte dataset

$$P' = P - \mu, \tag{4.11}$$

of the original Posebyte dataset $P$.

Generating a Conditional Posebyte from a subset of bits involves conditioning $N_p$ on the known bits' states. Expectation Maximization [16] on the new conditional distribution $N'_p$, a simple matter of taking the per-dimension mean over the new distribution, produces the most likely Posebyte. Finally, the values are rounded and clipped to be either a one or zero, representing true or false, respectively.

Conditional Posebytes provide a means of generating language queries for a subset of bits, but present a problem: the retrieved results will be holistically

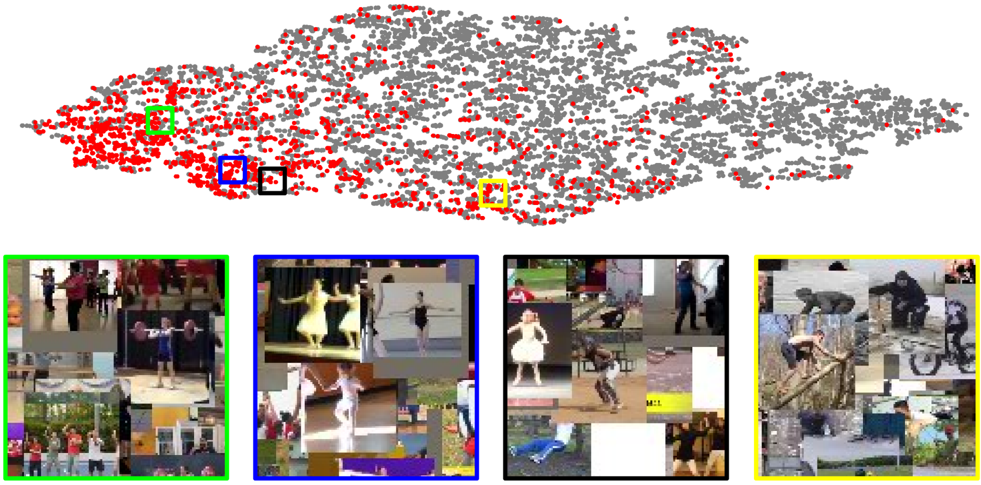


**Figure 4.4:** A t-SNE [55] embedding space visualization of the pose-aware embedding network defined by the Hamming distance metric over Posebyte language descriptors. The red and grey annotations indicate whether an image does or does not match the condition "right shoulder is bent," respectively. This condition is commonly true for pose configurations of (from left to right) standing with your elbows bent, standing with feet together and arms spread apart, arms and legs spread apart, and when in a hunched over position.

similar and thus will lack diversity in pose, as it captures only one mode of the desired priors. Figure 4.4 illustrates this problem, showing a few locations on the embedding space for when the "right shoulder is bent" bit is true. This issue stems from the original definition of the embedding space, i.e., any one location on the embedding space alludes to a pose defined in its entirety. Thus, once a Posebyte is embedded (conditioned or not), the Nearest Neighbours in the embedding space match the query maximally.

This problem is easily mitigated by, instead, sampling the conditional distribution. Sampling $N_p'$ multiple times produces a set of Posebytes each of which suffers from the same issue of local holistic similarity, but as a whole may produce variety.

While generating sampled Conditional Posebytes is fast, the fact that this process resides on the data-end of the model makes it cumbersome. Retrieving a set of images with variety involves the generation of sampled Conditional Posebytes, forwarding each of them through the network, and finally conducting a Nearest Neighbours search from their embedding destination.

**Query-Aware Masks** The pose-aware embedding network produces a unit hyper-sphere embedding space which co-locates holistically similar poses. Similar to the motivation of Conditional Posebytes, one may be interested in retrieving a set of results with a subset of pose conditions. As a consequence of being defined by a holistic pose metric, the current embedding space does not provide separation by condition. Inspired by [88], proposed here is a linear warping which collapses the unit hyper-sphere embedding space such that the entries of interest will lay nearest to the origin, simplifying the search space to a single point.

Figure 4.5 presents the proposed architecture for learning Query-Aware Masks. The embedding space learned by either stream of the image-language pose-aware embedding network is followed by a point-wise multiplication (a mask),

$$e'_{emb} = e_{emb} v_{mask}, \tag{4.12}$$

resulting in a warped embedding vector $e'_{emb}$, where $e_{emb}$ and $v_{mask}$ define an embedding vector and a linear warping mask, respectively. Each mask is specific to a language-primitive conjunction (a chosen subset of language-primitives and their states), and thus for each desired language-primitive conjunction a mask must be learned. A mask is represented by a vector with the same number of dimensions as the original embedding space. Each element of

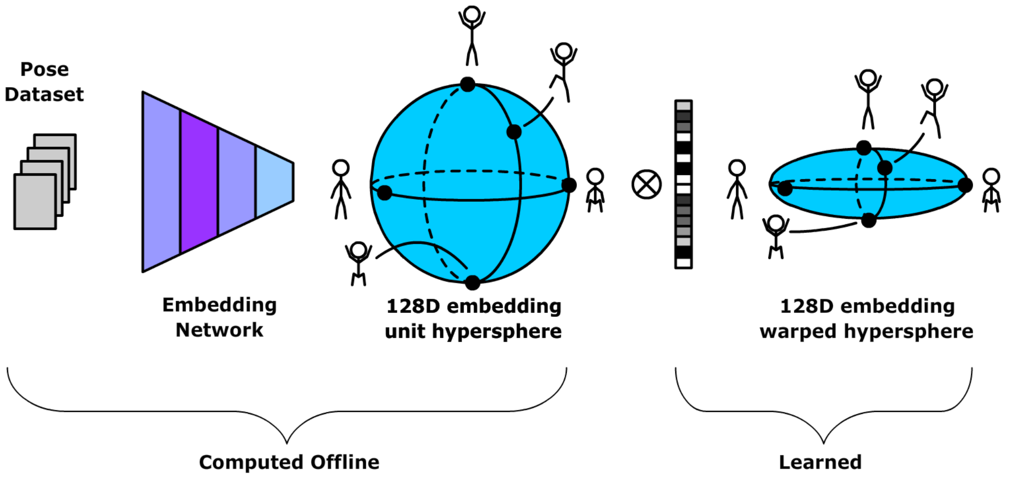

Pose Dataset
Embedding Network
128D embedding unit hypersphere
128D embedding warped hypersphere
Computed Offline
Learned

**Figure 4.5:** Conditional masks are used to linearly warp the embedding space such that the entries of interest lay closest to the origin. One mask is trained per language-primitive query subset, each of which is learned equally well from the embedding space generated by either the image or language stream of the image-language pose-aware embedding network. Each element of the mask vector is initially set to one to preserve the original embedding space, and is then free to change while training.

a mask is initially set to one to preserve the original embedding space, and is then free to change during training.

Query-Aware Masks operate directly on the embedding space, and thus the process of learning Query-Aware Masks does not require the original network. Instead, only a dataset of pre-computed embedding vectors is required.

Masks are learned using triplet rank loss, with the positive, and negative entries of a triplet containing, an embedding which matches the desired conditions, and one which does not, respectively. Rather than providing an anchor embedding, the anchor is instead the origin. That is, embeddings which correctly match the desired conditions should be re-mapped such that they are closer to the origin than those which do not. For example, one may be interested in images of people whose hands are above their head. The dataset of embedding vectors is then divided into the positive entries of “left hand is above head; right hand is above head,” and negative entries where neither or merely one of these conditions are met. Finally, the warping mask is regressed using triplet rank loss, leaving the original embedding space unaltered.

Warping the embedding space such that the relevant subspace is collapsed to the origin, simplifies the search process. Thus querying for any language-primitive subset becomes a matter of warping the embedding space, followed by conducting a Nearest Neighbours search starting at the origin.

Query-Aware Masks, unlike Conditional Posebytes, have the benefit of operating directly upon the embedding space, avoiding the need to forward any data through the network. In fact, unless novel instances are to be introduced into the dataset, the embedding network remains entirely unused. More so, where sampled Conditional Posebytes provide a set of prototype locations from where to start multiple queries, Query-Aware Masks simplify the query process by providing a single search point.

While Query-Aware Masks simplify the query process, generating the masks themselves presents a new challenge. For each language-primitive subset a new mask must be learned. For example, consider the set of first and second order language-primitive conjunctive queries. Covering every possible combination, the former would require 220 masks, while the latter would require more than 24,000 masks. In fact, as the order of conjunctions grows, the space of possible masks grows exponentially. More so, unlike Conditional Posebytes which are generated analytically and therefore almost instantly, Query-Aware Masks are solved numerically and thus learning a mask requires about one minute of training time.

# Chapter 5

# Experiments

This chapter introduces a novel pose similarity dataset, presents the details necessary for training pose-aware embedding networks, and concludes with a set of evaluations to demonstrate their efficacy.

## 5.1 Overview

This section presents the datasets and metrics used to train and evaluate the proposed models.

### 5.1.1 Datasets

For training pose-aware embedding networks only the dataset created for the purposes of this thesis was directly used. However, the base VGG-S network was pre-trained with the ImageNet dataset [17].

**The proposed dataset** This dataset was created for the purpose of training and evaluating pose-aware embedding networks across a variety of metrics. It is composed of approximately 24,000 16-frame image sequences, with accompanying annotations for 2D and 3D joint locations, as well as a set of language-primitives, i.e., Posebyte pose descriptors [72]. Also provided is a family of distance tables, describing the similarity between each dataset pair, over 2D and 3D Euclidean, Procrustean, and language-primitive Hamming metric distances.

**ImageNet** An in-the-wild dataset (not created from a staged setting) composed of over 14 million images feature 1,000 classes [17], including people, animals, vehicles, and architecture. It was created for the purpose of training and evaluating large-scale neural network models for the task of object recognition. This dataset was not used directly, but rather was used to pre-train the convolutional component of the pose-aware embedding networks.

**Kinetics** An in-the-wild large scale action recognition video dataset [41] with over 160,000 video clips, covering 400 unique actions. It was created for the purpose of training and evaluating large scale spatiotemporal neural networks for the task of action recognition. This dataset serves as the raw video data used for the construction of the dataset proposed in this thesis.

### 5.1.2 Evaluation Metrics

The efficacy of the proposed pose-aware embedding models are evaluated across two main experiments. First, these networks should learn to co-locate semantically similar images in the embedding space, and thus should act as effective similarity retrieval models. Second, two approaches for circumventing the limitations of holistically defined embedding spaces are demonstrated, with an evaluation of how the model can be successfully made to allow for querying with a subset of language-primitives (as opposed to the entire pose descriptor).

**Similarity Retrieval** To demonstrate the proposed models' efficacy to co-locate semantically similar images in the embedding space, two standard metrics are considered: hit@k, and mean distance@k.

Hit@k is a measure of whether the network has correctly learned to approximate the semantic ordering of the desired metric space. Each query is considered to be correctly ordered if at least one of its first $k$ retrievals is within the top-50 ground truth nearest entries, as expressed by,

$$hit_k(X) = \begin{cases} 1, & \text{if } |X \cap T_{50}| > 0 \\ 0, & \text{otherwise} \end{cases}, \tag{5.1}$$

where $X$ is a set of $k$ retrievals, and $T_{50}$ is the set of ground truth entries which are most similar to the respective query. Hit@k is measured across the entire set of queries, and is expressed as,

$$hit@k = \frac{1}{N} \sum_{i=1}^{N} hit_k(R^i_{1 \dots k}), \tag{5.2}$$

where $N$ is the number of queries, and $R^i_{1 \dots k}$ is the first $k$ retrievals for the $i^{th}$ query.

Distance@k measures the mean pose-descriptor distance between each query and its $k$ retrieved entries, with a smaller distance implying a closer metric space approximation. The distance is measured as,

$$distance_k(X) = \frac{1}{N} \sum_{i=1}^{N} \delta(X_i, q), \tag{5.3}$$

with $X_i$ being a retrieved entry, $q$ being the query, and $\delta$ being the distance as defined by the metric of which the network was intended to approximate. Distance@k is measured across the entire set of queries, and is expressed as,

$$distance@k = \frac{1}{N} \sum_{i=1}^{N} distance_k(R^i_{1 \dots k}), \tag{5.4}$$

where $N$ is the number of queries, and $R^i_{1...k}$ is the first $k$ retrievals for the $i^{th}$ query.

**Question-subset Queries** To demonstrate that the limitations presented by holistically defined embedding spaces may be circumvented using Conditional Posebytes and Query-Aware Masks, for the purpose of querying with language-primitive subsets, recall@k is considered. Given that the embedding space is holistically defined, similarity resides solely at a local level. Any one particular pose condition likely exist in vastly different overall poses, and thus each pose condition is spread across the entire embedding space in smaller holistically similar clusters. To understand this sparse distribution, recall@k measures model quality at a local level by calculating the percentage of correctly retrieved entries over the total number of retrievals, as expressed by,

$$recall_k(X) = \frac{|X \cap X_{tp}|}{|X|}, \tag{5.5}$$

where $X$ is a set of $k$ retrievals, $X_{tp}$ is the set of true-positives (retrievals matching the desired condition) in $X$, and $\cap$ indicates the intersection between the two sets. Specifically, this metric measures the number of correctly retrieved entries over the number of retrievals. Recall@k is measured across the entire set of queries, and is expressed as,

$$recall@k = \frac{1}{N}\sum_{i=1}^{N} recall_k(R_{1...k}) \tag{5.6}$$

where $N$ is the number of queries, and $R^i_{1...k}$ is the first $k$ retrievals for the $i^{th}$ query.

### 5.1.3 Training Details

Each pose-aware embedding network is trained using the dataset proposed in this thesis, and follows the data augmentation strategy and learning curriculum as outlined in [44].

**Pose-Aware Embedding Networks** Training was conducted using the triplet rank loss objective function, with a global learning rate of $10^{-2}$, which was reduced by $2 \times 10^{-1}$ every epoch for six epochs. The convolutional component of each network was initialized with the VGG-S [11] ImageNet [17] weights, easing the learning process by providing each network with a set of high quality image processing filters. Due to the modest amount of data, the learning rate for the convolutional layers were scaled by $10^{-1}$. These networks were trained using the backpropagation algorithm, with Nesterov accelerated gradient descent [3], a momentum of 0.9 and a weight decay rate of $10^{-5}$.

**Data Augmentation** Neural networks require a large amount of training data, and thus to artificially increase the number of training examples, the images were augmented online in such a way as to preserve their semantic meaning while increasing visual variety. They were randomly cropped by ±10%, and the entire batch was horizontally flipped with a probability of 50%.

**Batch Selection** Rather than having the network attempt to learn fine-grained pose similarity immediately, a task which may present some difficulties, the network is progressively given more challenging triplet examples over time, as previously outlined in (Section 4.2.1). Each iteration (one update step while training the model), the network is presented with an anchor, five similar examples, and 105 dissimilar examples. The anchor selection is shuffled such that the network sees each dataset entry as an anchor once per epoch. The positive examples are selected from the first 30 similar ground truth dataset entries, and the negative examples are selected from the remaining set. To ensure that the network progressively learns to identify more fine-grained pose similarity over time, after each epoch 3,000 of the most dissimilar examples are excluded from the selection process, until there are at most 1,000 negative entries to select from.

**Query-Aware Masks** Each mask is modeled by a 128-dimensional pointwise multiplication layer. Embedding the entire training dataset of images offline, a new Query-Aware Mask is learned for an question-subset using triplet rank loss. However, instead of operating on images, Query-Aware Masks operate directly on the embedding space, and thus triplets are composed of three embedding vectors. Each mask is defined by a designated set of question-subsets and thus for each mask the anchor is a well established point, i.e., the origin, with similar and dissimilar embedding vectors being randomly selected. Masks are trained with the intention of collapsing the relevant subspace, placing the matching entries closest to the origin, and thus easing the query process.

The model is composed of a single point-wise multiplication layer and was trained using the backpropagation algorithm, with Nesterov accelerated gradient descent, a momentum factor of 0.9, a weight decay rate of $10^{-5}$, and a global learning rate of 1.0. Prior to training, the mask is initialized such that each component of the 128-dimensional vector is set to one, keeping the original embedding space intact and allowing the network to learn a linear deformation.

For query subsets beyond a single language-primitive, it was discovered that it is important to provide a balance of the possible combinations of negative classes. For example, while training a mask with the intention of collapsing the subspace responsible for "left knee bent; right knee bent," the negative class would include "left knee not bent; right knee bent," "left knee bent; right knee not bent" and "left knee not bent; right knee not bent."

Given that Query-Aware Masks essentially learn a new layer on top of the multi-modal image-language embedding network, either the image or language stream can be used to generate the embeddings, to similar effect.

## 5.2 Human Pose Similarity Dataset

This section outlines the motivation for creating a new dataset, and describes the steps taken to construct the proposed human pose annotations.

### 5.2.1 Motivation

This thesis proposes a new dataset designed for the purpose of training pose-aware embedding networks. Following the process outlined in [44], proposed here is a dataset constructed using the Kinetics video dataset [41].

While there is an existing pose similarity dataset [44], this thesis proposes to construct a new dataset for the purpose of seeding future work into the exploration of video embedding metrics. This thesis does not explore embedding spaces outside of image data, however by providing a large amount of in-the-wild per-frame joint annotations which conform to the common practices of image [44, 62] and video networks [33, 87], this dataset creates a common baseline for image and video embedding network architectures.

### 5.2.2 Pose Descriptor Construction

For the purpose of training pose-aware embedding networks, proposed here is a process for constructing a large annotation dataset of human pose descriptors. The three types of pose descriptors considered are 2D pixel and 3D metric joint locations, as well as a set of language-primitives [72]. Figure 5.1 demonstrates a selection of example images, and the various kinds of provided annotations.

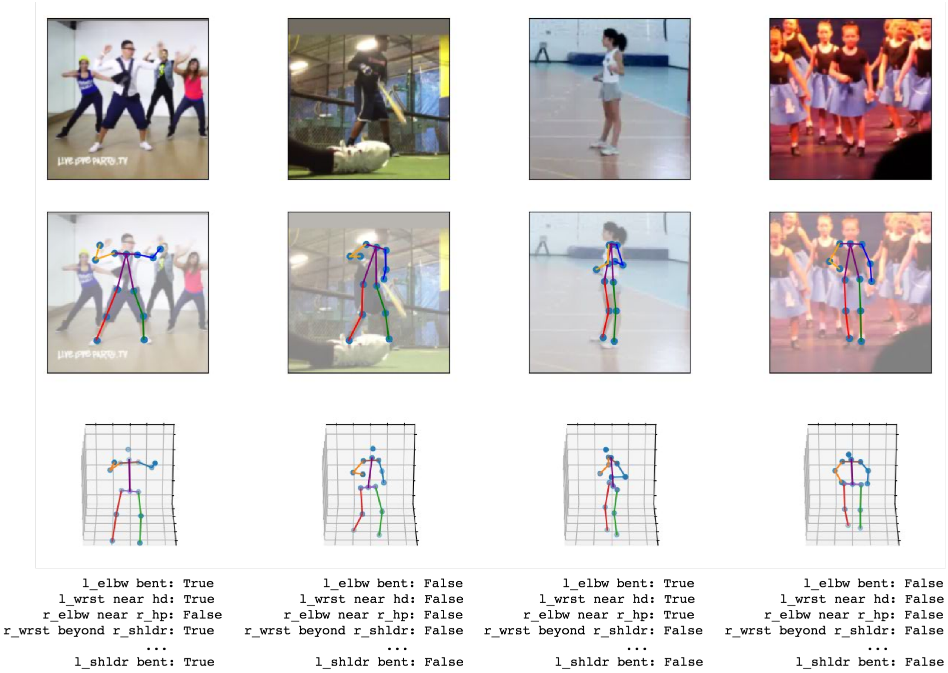


**Figure 5.1:** A depiction of the proposed dataset. It is composed of a mapping between images, 2D, 3D, and language-primitive (Posebyte [72]) descriptors.

### 2D Skeletal Pose Descriptors

Framing human pose as a set of 2D joint annotations is by far the most common approach to addressing human pose estimation [86, 9, 96]. These annotations generally come in the form of pixel coordinates, such as the head, hands, and knees. 2D human pose estimation has been shown to be a relatively well addressed problem, with modern CNN approaches performing strongly on a variety of community benchmarks. Taking advantage of the high quality nature of contemporary pose estimators, OpenPose [9, 96] is used to generate per-frame multi-person annotations for the Kinetics dataset, although any number of modern pose estimators would likely have worked equally well. Proposed here is a process to temporally link and rectify per-frame joint annotations across a set of video sequences.

**Joint production** Taking advantage of the abundance of raw video data provided by the Kinetics action recognition video dataset, OpenPose was used to generate a large scale dataset of per-frame joint annotations. For each joint location OpenPose provides a confidence value, and thus to ensure that low quality predictions are suppressed, only poses which have an acceptable confidence level, of at least 12 of the 13 relevant joints, are retained. Any joint was considered acceptably confident if the estimator reported a value of at least 10%, as there did not appear to be any empirical difference in joint quality beyond this suggested value.

**Temporal linking** Although OpenPose does not directly provide an inter-frame pose relationship, the generated poses were remarkably consistent between neighbouring frames. Leveraging the relatively high frame rates of the video, a simple Intersection over Union (IoU) [25],

$$IoU(A, B) = \frac{A \cap B}{A \cup B}, \tag{5.7}$$

of the minimum pose-encapsulating bounding boxes $A$ and $B$, between neighbouring frames, was used to determine whether any two poses were derived from the same actor. Any two poses between neighbouring frames were considered to be the same actor if their bounding box overlapping IoU score was larger than 70%, an empirically determined value. In the case that there was a conflict between multiple poses, the largest IoU was determined to be the same actor.

While this process worked well, it relies on consistent per-frame pose predictions, a lack of large occlusions, and actors remaining in the scene for the duration of the clip. Tracking actors across a scene is a challenging problem in-and-of-itself, and thus following common practice [47], these artificial separated sequences are treated as distinct actors.

The following rectification process takes advantage of the temporal component to ensure a strong consistency between inter-frame joints, and thus a minimum number of each joint is required to be present, as well as additional padding frames (two before and two after) to avoid having to handle the first and last frame edge-case. While this thesis does not conduct video based experiments, sequences with less than 16 frames (20 when padding is considered) were discarded. Maintaining sequences of at least 16 frames, a common choice for spatiotemporal convolutional neural networks [33, 87, 62], allows for potential future work into video based pose-aware embedding networks.

**Pose canonicalization** Following the process outlined in [44], each image is padded with an additional 20% of image space to allow for augmentation by random spatial cropping, during the training process. The human pose falls directly in the middle of a square region, within the padded area. The final product is a set of video clips that are relative to the actor.

Often enough the pose estimator produces errors, such as joints which tend to oscillate between two locations. To reduce noise in the final square-cropped video sequences, each frame's bounding square is averaged over its two left and right temporal neighbours. This smoothing produces video sequences which are more stable.

Finally, each cropped video clip is spatially resized to 144x144 pixels, with the relevant poses being scaled proportionally with the video.

**Joint hysteresis** While OpenPose is relatively consistent across frames, in some cases the per-joint confidence values tend to fluctuate, especially in situations of large motion, noisy video, and partial occlusions. Proposed here is an approach to increase the per-joint confidence values by leveraging the temporal component of the video data.

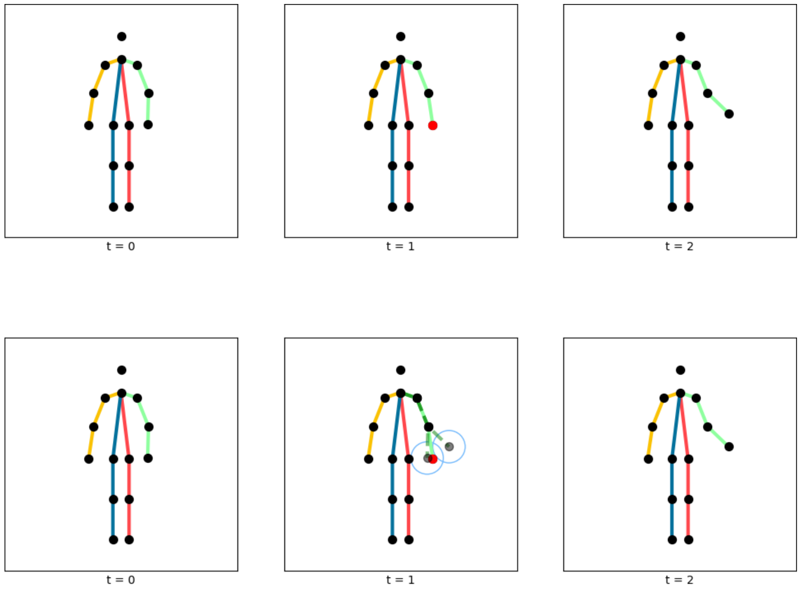


**Figure 5.2:** An example of joint hysteresis. The top and bottom row show the same raw and rectified annotation sequence, respectively. In the top row, the middle frame contains a low confidence “left wrist” joint, marked in red. The bottom row indicates that this same joint is indeed valid, as it falls within a close proximity to the same joint within a neighbouring frame, as marked by the blue circle.

For any joint with a confidence value lower than 10%, its first and second order temporal-neighbours are considered. If any one neighbour exists within a 10 pixel Euclidean distance from the current joint, then the current joint is marked as confident, as depicted in Figure 5.2.

The choice of 10 pixels, as well as considering only the first and second order neighbours was empirically determined. Any joint whose confidence value could not be rectified was discarded.

**Joint denoising** While OpenPose's estimations are mostly consistent between frames, they can sometimes suffer from noisy localization, mainly in the form of joints bouncing back and forth between multiple spatial locations. This

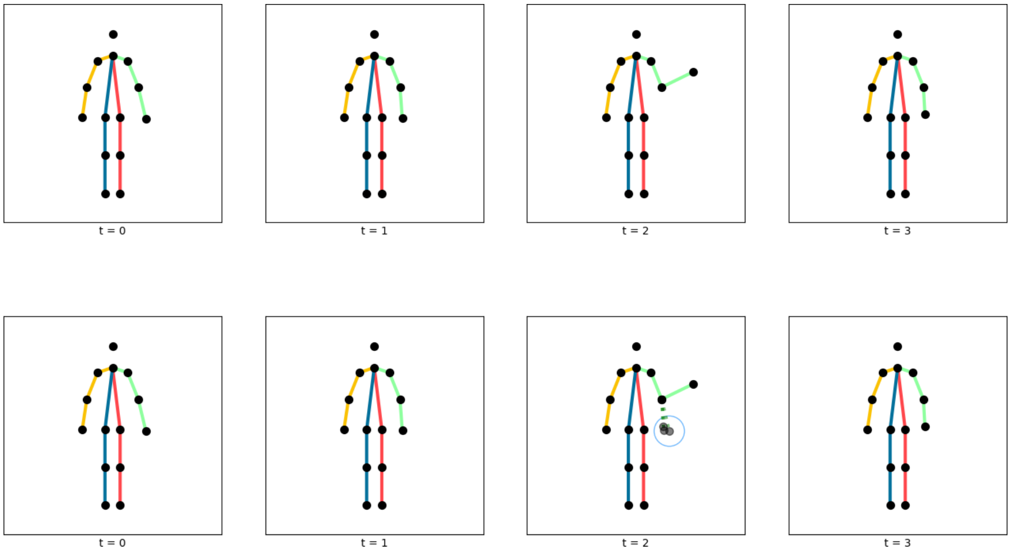


**Figure 5.3:** The top and bottom row show the original and rectified annotation sequence, respectively. The top row presents an example of a spurious limb. Given that the remaining joints have built a consensus on acceptable joint locations, the spurious limb is replaced with its neighbour's.

is likely a consequence of OpenPose having to make a hard decision between multiple noisy, yet relatively high-confidence probability regions. Proposed here is a consensus based approach to replace noisy joints.

To identify and replace noisy joints, a window of size four is run forward and backward across the frame sequence, as depicted in Figure 5.3.

In the case that the current frame's two previous and one following neighbours' relevant joints maintain a certain amount of spatial consistency, i.e., come to a consensus, then the current frame's joint may be assessed to determine if it is behaving spuriously. A consensus is formed if the designated neighbour frames spatially agree on a joint location, within 10 pixels of the earliest considered frame. If the current frame's joint is not also within this 10 pixel threshold, then the relevant limb is replaced with its earliest neighbour's

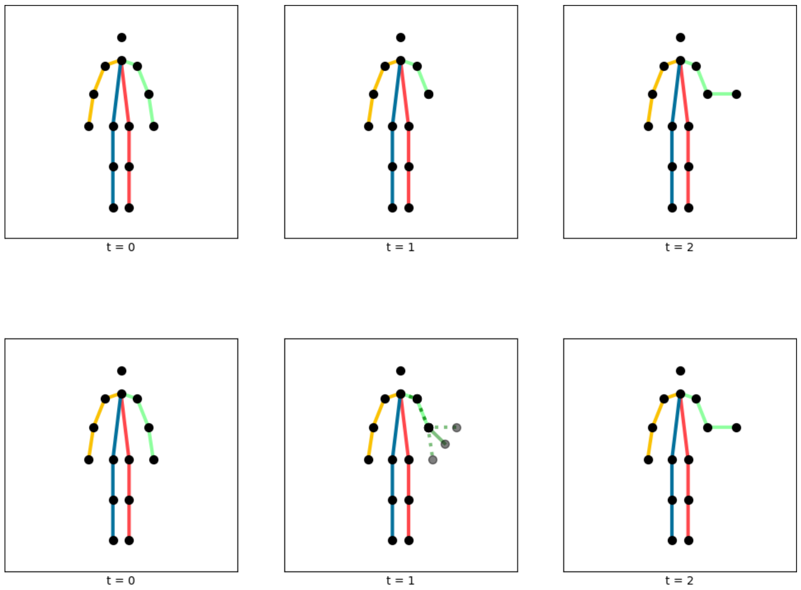


**Figure 5.4:** Missing joints are replaced by their nearest temporal neighbour or are interpolated between its two nearest neighbours. The top row contains three frames from the original sequence, where there is a missing “left wrist” joint. The bottom row contains the rectified sequence, with the missing joint being interpolated by its neighbouring frames.

limb. In the case that no consensus is formed, the frame is skipped as there is no way to confirm if the joint is indeed spurious.

**Joint hallucination** During joint production and hysteresis, joints may have either not been discovered by OpenPose, or discarded due to low confidence values. To complete each pose annotation, the missing joints are hallucinated.

Missing joints are replaced by their nearest existing temporal neighbour. In the case that there are two temporally equidistant joints, i.e., a valid joint exists at both $+t$ and $-t$ frames, then the new joint is simply the average between the two source joints.

This process produces approximately 49,000, relatively clean, and fully annotated human joint video sequences. While it would be tempting to use the entire 49,000 joint annotated video clips it is the case the same person may have artificially been split into multiple clips, due to noise, occlusion, or exiting and re-entering the frame. To ensure visual variety, approximately half of the video clips are removed. In a round robin fashion, a clip from each video was randomly discarded, until the final clip count was approximately 24,000.

Example produced 2D joint annotations are presented in Figure 5.1.

### 3D Skeletal Pose Descriptors

Constructing an in-the-wild 3D joint annotation dataset presents a significantly more difficult challenge when compared to its 2D counterpart, due to the innate inability of human annotators to correctly judge accurate spatial distances, thus 3D joint annotation datasets are far less common. Much like how 2D joint annotations provide pixel coordinates, 3D joint annotations provide metric coordinates for each joint. The two most successful approaches to estimate 3D joint locations are: direct 3D joint inference [49, 50] from raw pixel data, and 3D joint inference from 2D joint annotations, referred to as "lifting" [106, 85]. Taking advantage of the previously generated 2D joint annotations, and an off-the-shelf state-of-the-art lifting model [106], an equivalent 3D joint annotation dataset is generated. One major distinction from the 2D joint annotation generation process is that [106] operates spatiotemporally, and thus inherently takes advantage of the temporal component of the pose data.

**Joint production** The 3D human joint annotations are produced by "lifting" the existing 2D joints. Lifting involves estimating the most likely 3D joint configuration, given a set of 2D annotations.

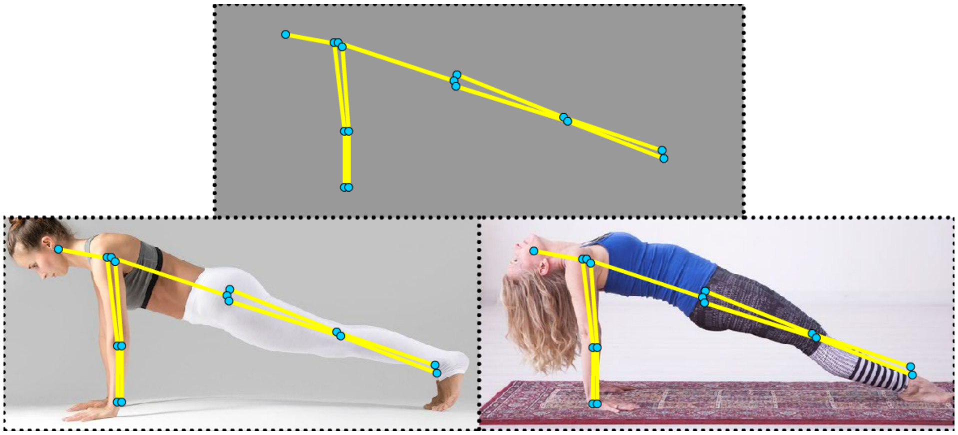


**Figure 5.5:** A depiction of 3D pose ambiguity when lifting 2D pose annotations. There are two common methods for generating 3D pose annotations: direct 3D joint regression, and lifting 2D joints to 3D. Lifting operates on 2D joint annotations, independent of the original image from which the 2D joints were derived. Top: an example of 2D joint annotations. Left and Right: two contrasting human pose images that closely match the 2D joint annotations, illustrating one possible source of 3D lifting ambiguity.

Treating each 3D pose in the Human3.6 MoCap (motion capture) [37] annotation dataset as an over-complete set of basis poses, [106] computes a linear combination to produce a new pose. Block coordinate descent, is used to update either the pose-coefficients, rotation, or translation. Coefficient hyper-parameters $\alpha$ and $\beta$ are set to 0.5 and 20.0, enforcing small model parameters and temporally smooth pose-coefficients, respectively. $\gamma$ is set to 2.0 and enforces rotational temporal smoothness. Each pose is regressed for a maximum of 10 iterations or until it has converged to an error of at least $10^{-4}$.

Much like 2D pose estimation, 3D pose estimation is error prone. The most common error involves poses being flipped about their joint depth positions, with respect to the direction perpendicular to the image plane. That is, where

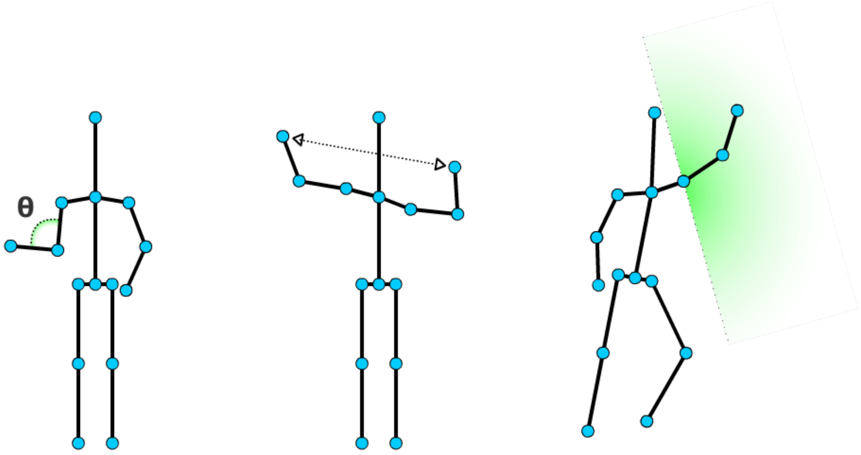


**Figure 5.6:** A depiction of the three kinds of Posebits, originally described in [72]. A Posebit is a language-primitive pose descriptor that indicates if a particular pose condition is true or false. Left: A joint is considered bent if its angle is beyond the relevant threshold angle. Middle: A pair of joints are considered far if they are further apart than their relevant distance threshold. Right: A joint is considered to be beyond another if the distance between said joint and the torso is further than a relative joint and the torso.

the left wrist joint should be closer to the camera than the right, under this effect the generated pose will suggest the opposite. To illustrate this kind of error, Figure 5.5 demonstrates the many-to-one ambiguity inherit to 3D pose lifting. Further, the lifted 3D joints also inherit any localization error produced during the 2D pose estimation step. While there is bound to be some error, the dataset was clean enough for the purpose of training 3D pose-aware embedding networks.

### Language-primitive Pose Descriptors

Language provides a natural way of communicating the configuration of a human's pose, yet is severely under explored with few exceptions [72, 63, 51]. While joint localization has been largely successful, given the desired goal of describing an actor's pose, joint localization is certainly overly pedantic. Pose is rarely communicated in terms of pixel locations or metric coordinates, but rather in terms of relative conditions. For example, a bank robber is unlikely to demand of his victims to “Put your hands 0.5 meters above your head!”

As a more natural alternative, this work explores the use of binary language-primitives in the form of Posebits [72]. Each Posebit describes a single condition, which translates to a simple natural language Boolean statement. For example, the bank robber, with the advent of Posebit language-primitives, may renew his demand of his victims, “Put your left hand above your head! Put your right hand above your head!” In the same way that a set of joints define an entire pose skeleton, a set of Posebits define an entire Posebyte pose descriptor. Posebits are easily derived by either asking annotators [72] to indicate if a specific pose condition is true or false when provided an image containing a human pose, or automatically from 3D joint annotation datasets [72]. Thus this thesis proposes to exploit the previously generated 3D joint annotation dataset to construct a Posebyte pose descriptor dataset.

There are three kinds of Posebits: joint angle bits, which specify whether a joint is bent; joint distance bits, which specify whether a pair of joints are far apart; and joint relative-distance bits, which specify, for a pair of joints, whether the first is further away from the torso-center than the second.

**Joint angle bits** These bits indicate whether a joint is bent beyond some threshold, as illustrated in Figure 5.6. For example, a joint angle bit set to one may describe “right knee is bent,” and set to zero would then describe “right knee is not bent.” The threshold angle for which a bit is set is defined empirically, on a bit-by-bit basis.

**Joint distance bits** These bits indicate whether a pair of joints are far away from each other beyond some threshold, as illustrated in Figure 5.6. For example, a distance bit set to one may describe “left wrist is far from right wrist,” and set to zero would then describe “left wrist is not far from right wrist.” The threshold distance for which a bit is set is defined empirically, on a bit-by-bit basis.

**Joint relative-distance bits** These bits indicate whether a joint is beyond its pair relative to the torso-center, as illustrated in Figure 5.6. Determining the bit’s state is a simple matter of calculating whether a joint is in front of the plane defined by the vector extending from the torso-center to the relative-joint. An example relative-distance bit set to one may describe “right hand is above head,” and set to zero would then describe “right hand is not above head.” However, the term “above” is misleading as Posebits are camera invariant, giving little meaning to relative positioning terms, such as “above,” “below,” “in-front,” and “behind.” The alternative terminology, suggested for the purpose of this thesis, is to consolidate the set of relative positioning terms with the single statement “beyond.” For example, “right hand is above head” will be restated as “right hand is beyond head,” which concisely describes all scenarios, independent of their orientation with respect to both the ground plane and the camera. In contrast to the other Posebit types, joint relative-distance bits do not require a threshold, as the joint is either in front of or behind the normal plane.

Posebits are derived from the 3D joint annotations, which were lifted from the 2D joint annotations, which themselves were estimated from image data, and thus any error in either the 2D or 3D joint annotations are likely inherited. More so, angle and distance Posebits are defined by hard thresholds, any noisy joints residing near the threshold may be prone to bit inversion. Similarly, any noisy joint near the relative-distance normal plane may also be incorrectly labeled. Defining hard thresholds may also present visual ambiguity. For example, if the threshold for when an elbow joint is considered bent is defined as 150°, then an elbow bent 149° and 151° are likely visually indistinguishable yet have contrasting bit values.

## 5.3 Evaluation

This section presents an empirical evaluation of pose-aware embedding networks. Specifically, these networks are assessed on their ability to: co-localize similar poses over a variety of metric spaces, and their capacity to inherently factorize the embedding space such that question-subset queries are made possible.

### 5.3.1 Retrieval

To determine if the pose-aware embedding networks have successfully approximated the semantic ordering of their respective metric spaces, each model is evaluated on its ability to correctly retrieve semantically similar entries. To put the retrieval quality into context, two baselines are provided: the base VGG-S network's fc7 feature [11], and random chance.

The proposed pose-aware embedding networks are a modification of the VGG-S Network [11]. To demonstrate that these models learn something distinct from their predecessor, the penultimate fc7 feature of the VGG-S

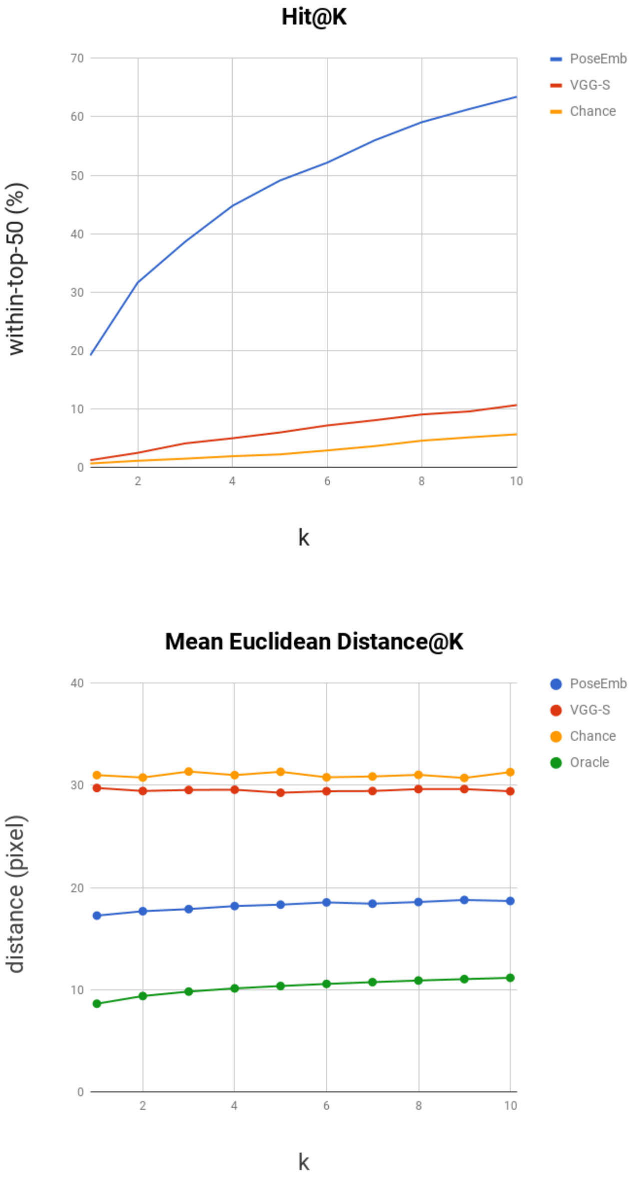


**Figure 5.7:** A quantitative evaluation of the 2D pose-aware embedding network (PoseEmb) defined by the mean-per joint Euclidean pixel distance metric. Three baselines are provided: the VGG-S network's fc7 feature, random chance, and an oracle. The oracle baseline indicates the best possible answer for any query.

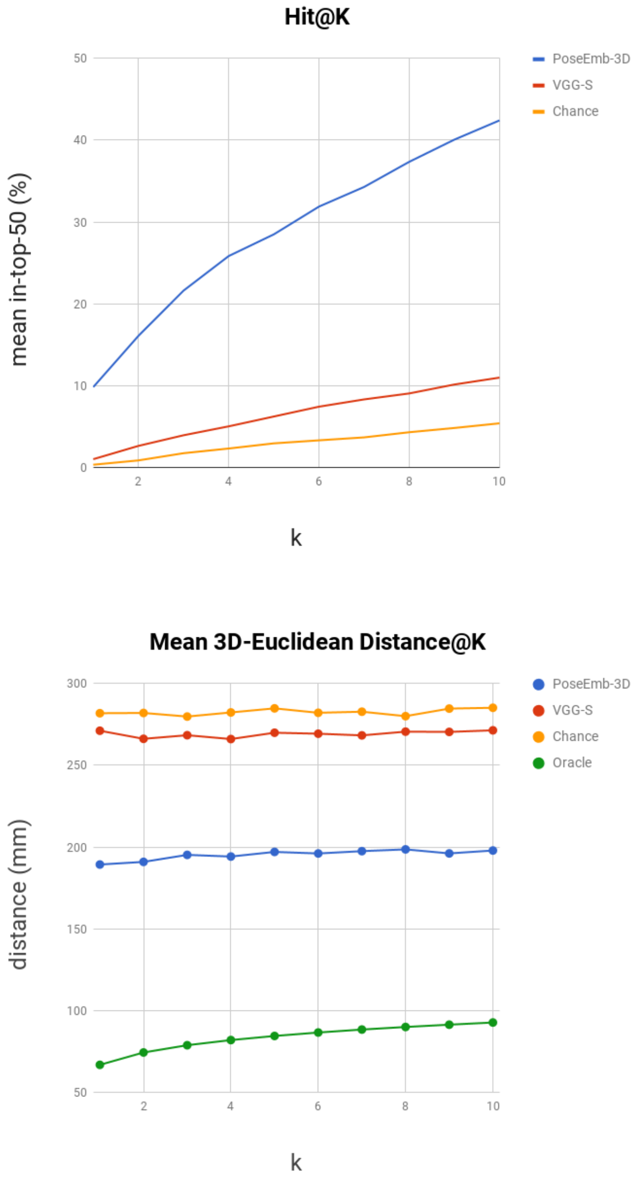


**Figure 5.8:** A quantitative evaluation of the 3D pose-aware embedding network (PoseEmb-3D) defined by the mean-per joint Euclidean metric distance. Three baselines are provided: the VGG-S network's fc7 feature, random chance, and an oracle. The oracle baseline indicates the best possible answer for any query.

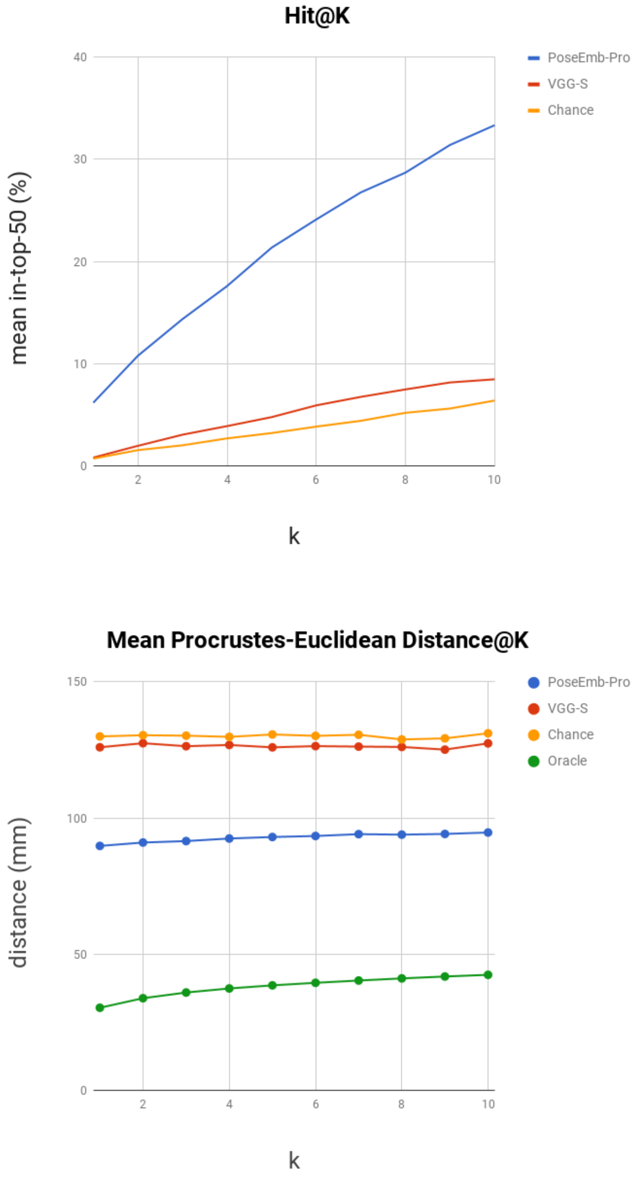


**Figure 5.9:** A quantitative evaluation of the Procrustes pose-aware embedding network (PoseEmb-Pro) defined by the mean-per joint Euclidean Procrustes metric distance. Three baselines are provided: the VGG-S network's fc7 feature, random chance, and an oracle. The oracle baseline indicates the best possible answer for any query.

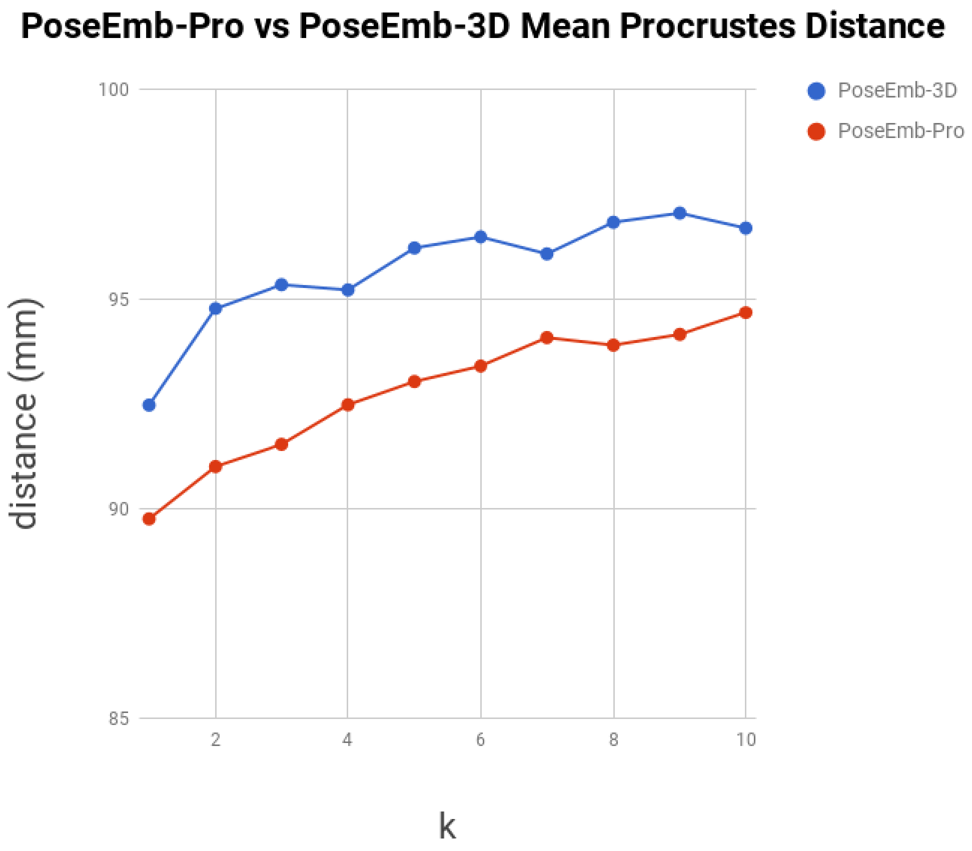


**Figure 5.10:** A contrasting quantitative evaluation of both the PoseEmb-3D and PoseEmb-Pro pose-aware embedding networks. Both PoseEmb-Pro and PoseEmb-3D operate within a metric distance space, and thus, provided here is a comparison between the PoseEmb-Pro and its camera-variant PoseEmb-3D counter part. To provide a common benchmark, both model's distance@k is measured using the Procrustes distance metric.

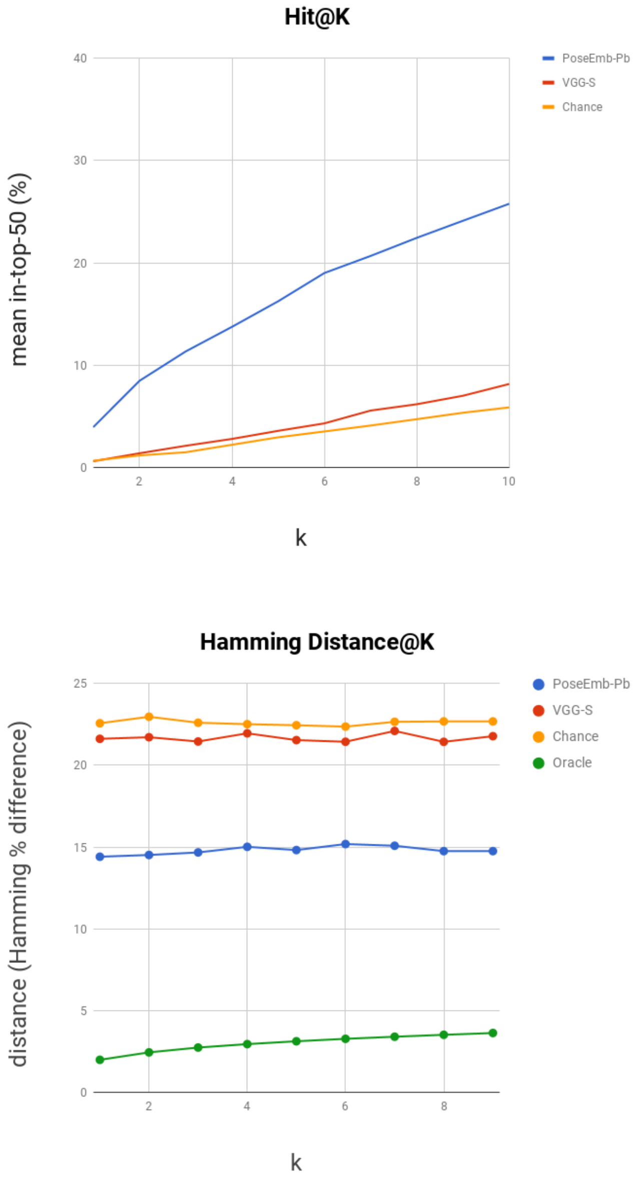


**Figure 5.11:** A quantitative evaluation of the Posebyte pose-aware embedding network (PoseEmb-Pb) defined by the mean Posebyte Hamming distance metric. Three baselines are provided: the VGG-S network's fc7 feature, random chance, and an oracle. The oracle baseline indicates the best possible answer for any query.

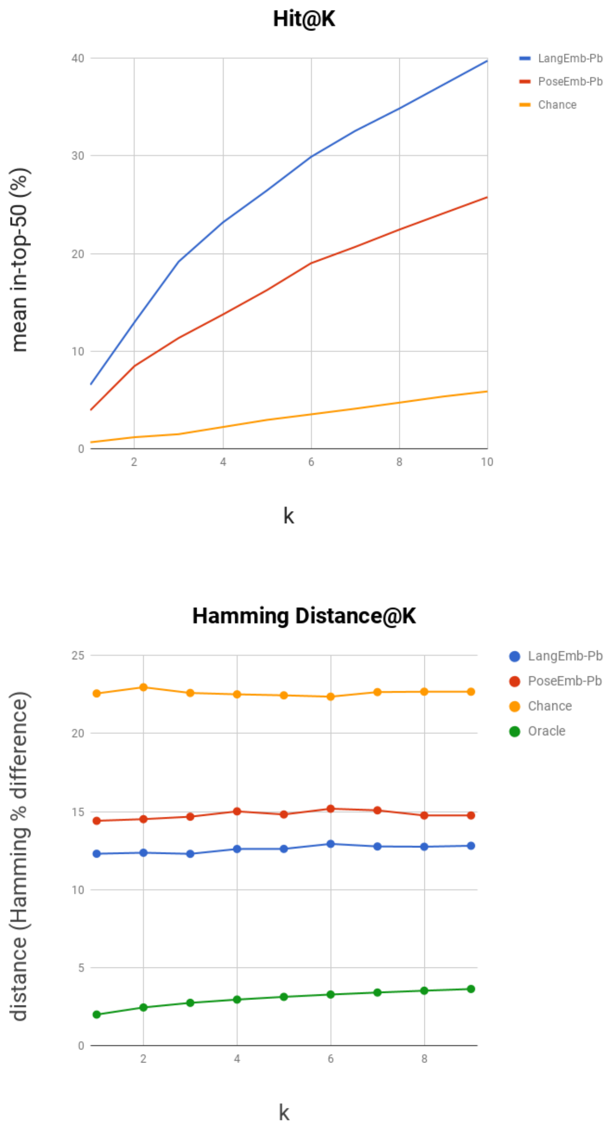


**Figure 5.12:** A quantitative evaluation of the language stream of the image-language pose-aware embedding network (LangEmb-Pb). The embedding space was filled with test set image embeddings, and the queries were composed of validation set Posebyte embeddings. Three baselines are provided: the image stream of the same network (PoseEmb-Pb), random chance, and an oracle. The oracle baseline indicates the best possible answer for any query.

Network is used as a baseline.

Finally, random chance is used to demonstrate that the proposed models have learned something beyond any bias which may be present within the dataset.

**PoseEmb** Figure 5.7 presents a quantitative evaluation for hit@k and mean distance@k. The PoseEmbedding Network (PoseEmb), trained over the 2D pixel distance metric, vastly improves over its base network. Trained for the purpose of object recognition [11], the VGG-S Network learns a set of filters useful for predicting the class of the main object present within an image. These networks show a tendency to cluster images by similarity, as demonstrated by their intermediate representations [1, 98, 58, 105, 2, 55]. Fine-grained pose similarity may be a more challenging task for this model, as a VGG-S network not specifically fine-tuned for pose may tend to focus on concepts which are helpful for the original task of object recognition.

**PoseEmb-3D** Figure 5.8 presents a quantitative evaluation for hit@k and mean distance@k, for the pose-aware embedding network defined by 3D joint similarity. One might expect the 3D data to allow for more discrimination in pose, resulting in a hit@k superior to that of 2D, yet the opposite is observed. One possible interpretation is that 3D human pose localization is challenging, both explicitly and implicitly, and thus the model is susceptible to depth ambiguities and strong projective geometry side effects, such as foreshortening. One other possible source of concern may be that the ground truth data is a lifted estimation of 2D joint annotations (which themselves are estimates).

**PoseEmb-Pro** Figure 5.9 presents a quantitative evaluation for hit@k and mean distance@k, for the pose-aware embedding network defined by Procrustes Transform aligned 3D joint similarity. While it appears that PoseEmb-Pro is a significantly less effective retrieval model, when compared to PoseEmb-3D,

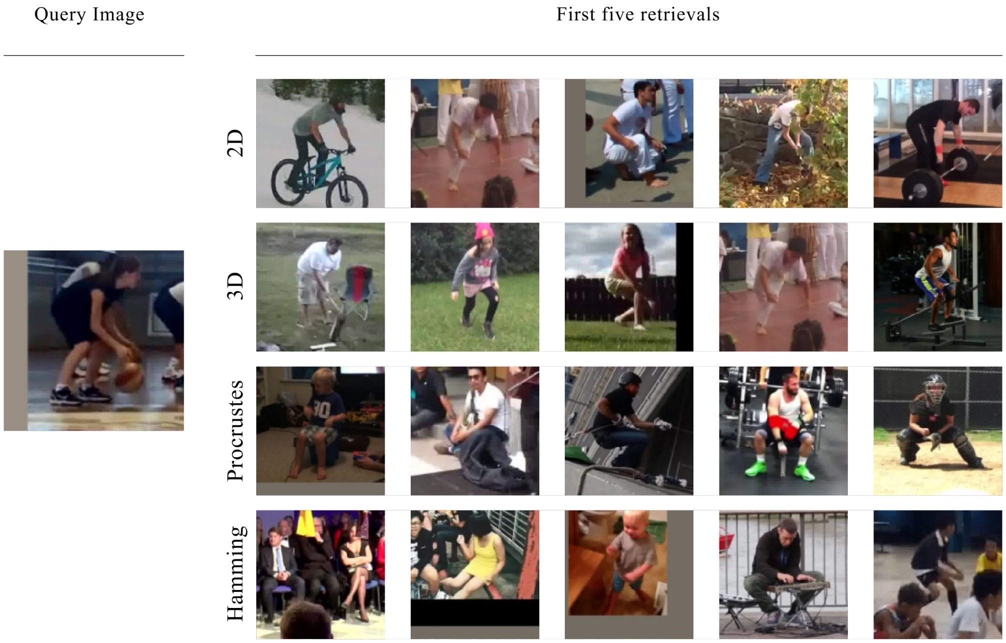


**Figure 5.13:** An example image query across all four networks. Each network was trained to respect a desired metric, including (from top to bottom) 2D mean per-joint pixel distance, 3D mean per-joint metric distance, 3D mean per-joint Procrustes aligned metric distance, and Posebyte Hamming distance. It should be noted that unlike the 2D and 3D networks, the Procrustes and Hamming distance networks are camera invariant.

Figure 5.10 demonstrates that when evaluated under the Procrustes distance metric, PoseEmb-Pro is able to retrieve entries which are more similar, by approximately 3mm on average. The PoseEmb-3D performs fairly well under the Procrustes distance metric, but is inherently restricted to similar poses which also roughly match the respective camera position and orientation. On the other hand, PoseEmb-Pro is not bound to any constrains outside of human pose, and thus has access to a wider variety of poses from which to retrieve. The lower hit@k values can therefore be attributed to the fact that PoseEmb-Pro is expected to find more fine-grained poses for each query, than is required of PoseEmb-3D.

**PoseEmb-Pb** Figure 5.11 presents a quantitative evaluation for hit@k and mean distance@k, for the pose-aware embedding network defined by the Hamming distance over Posebyte vectors. Similar to PoseEmb-Pro, PoseEmb-Pb is camera invariant, and is therefore expected to find more fine-grained poses for each query than that of camera variant models, explaining its lower hit@k.

**LangEmb-Pb** Providing a second mode for querying, the language stream of the image-language pose-aware embedding network is evaluated quantitatively in Figure 5.12. Interestingly, querying with Posebytes improves over images by 3% to 14% under the hit@k metric, and is 2% more similar on average under the distance@k metric. One possible reason for this improvement could be the extant nature of Posebytes. Unlike images, which obfuscate the underlying pose with pixel values, Posebytes clearly state the pose configuration.

Retrieval examples, for all four image embedding networks, are illustrated in Figure 5.13.

### 5.3.2 Question Subset Queries

The proposed pose-aware embedding networks are regressed such that holistically similar poses are co-located in the embedding space. For example, given a location on the embedding surface where an actor is standing in a Y-shape (upright with their arms stretched over their head), a Nearest Neighbours search would yield images of actors which are almost entirely in the same posture.

This means that any Posebyte language-primitive query (or image query) will retrieve results with little variety. This may not be desirable, as one may want to search for a subset of language-primitives, such as "right elbow bent; left elbow bent," with little care for how the remainder of the pose is arranged.

To allow for queries composed of a subset of the original language-primitives, two approaches are proposed: Conditional Posebytes and Query-Aware Masks. Both approaches are non-intrusive, in that the parameters of the model remain unaltered. While two distinct approaches are presented, they address the problem from the opposite ends of the model: the data side, and the embedding side, respectively.

**Conditional Posebytes** A Conditional Posebyte is an artificially generated language-descriptor. Assuming that Posebytes are defined under a multinomial Gaussian distribution, generating a Conditional Posebyte becomes a simple matter of conditioning the Gaussian for any subset of language-primitive priors. The remaining bits can be resolved by either maximum likelihood estimation [16] (taking the per-feature mean over the new distribution), or can be sampled directly from the new conditional Gaussian distribution.

Figure 5.14 (top) outlines a comparison between Conditional Posebytes and querying using the validation set (Mean Query) Posebytes. Each query was evaluated against its nearest $k$ neighbours, with a retrieved entry being

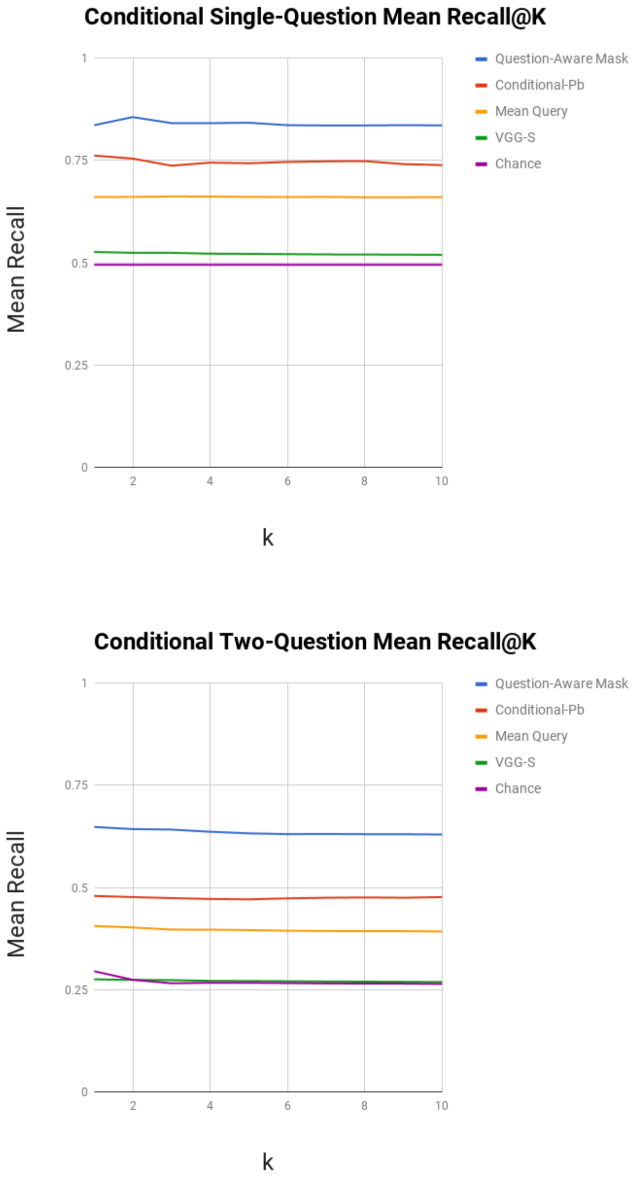


**Figure 5.14:** A quantitative evaluation of single-question queries (top) and two-question queries (bottom). As illustrated, Query-Aware Masks and Conditional Posebytes significantly outperform queries of the existing Posebyte validation set (Mean Query).

considered a match if it contains the correct value at the corresponding Posebit.

Conditional Posebytes significantly outperform querying with the Posebyte validation set (Mean Query) by 11% to 15%. With the understanding that the embedding space is defined holistically over the entire pose, one possible reason for this stark quantitative difference may be that Conditional Posebytes contain a combination of bit-values which are maximally likely for the desired bit and its condition, and thus when embedded would most likely be co-located with types of poses where the bit-of-interest is generally in the desired state. On the other hand, the validation set Posebytes will contain less likely Posebit configurations. This means that when they are embedded, the bit-of-interest does not necessarily have to be in the correct state so long as the Posebyte generally matches its neighbours overall. Similarly, Figure 5.14 (bottom) demonstrates that two-question Conditional Posebytes maintain a large margin over a two-question Mean Query.

Maximum likelihood estimation of Conditional Posebytes limits the potential variety of retrieved entries, as each Posebyte is mapped to a single location on the embedding surface retrieving only holistically similar entries. Sampling the conditional Gaussian distribution provides a mechanism for selecting multiple sub-optimal (yet highly-likely) bit configurations, each of which will map to a unique location on the embedding surface. While Figure 5.15 illustrates the benefits of sampling Conditional Posebytes for new-found visual variety, Table 5.1 outlines the consequences, as any excursion away from the maximum likely Posebyte leads to a decrease in accuracy. This is to be expected, as sampled Conditional Posebytes may end up mapping to locations on the embedding surface where they maximally match their surroundings without the bit of interest being present.

| **Mahalanobis distance** | **mean recall@1** |
|---|---|
| 0.1 | 73.1 |
| 1.0 | 71.1 |
| 10.0 | 64.7 |
| **Baselines** | **mean recall@1** |
| Conditional Pb | 76.2 |
| Mean Query | 66.1 |

**Table 5.1:** Outward sampling of conditional Gaussian distributions leads to a decrease in accuracy. The Mahalanobis distance is the multinomial equivalent of the standard deviation. Each distribution was sampled 25 times at the designated Mahalanobis distance, with a successful match being considered true if the retrieved entry's corresponding Posebit is set to the desired state.

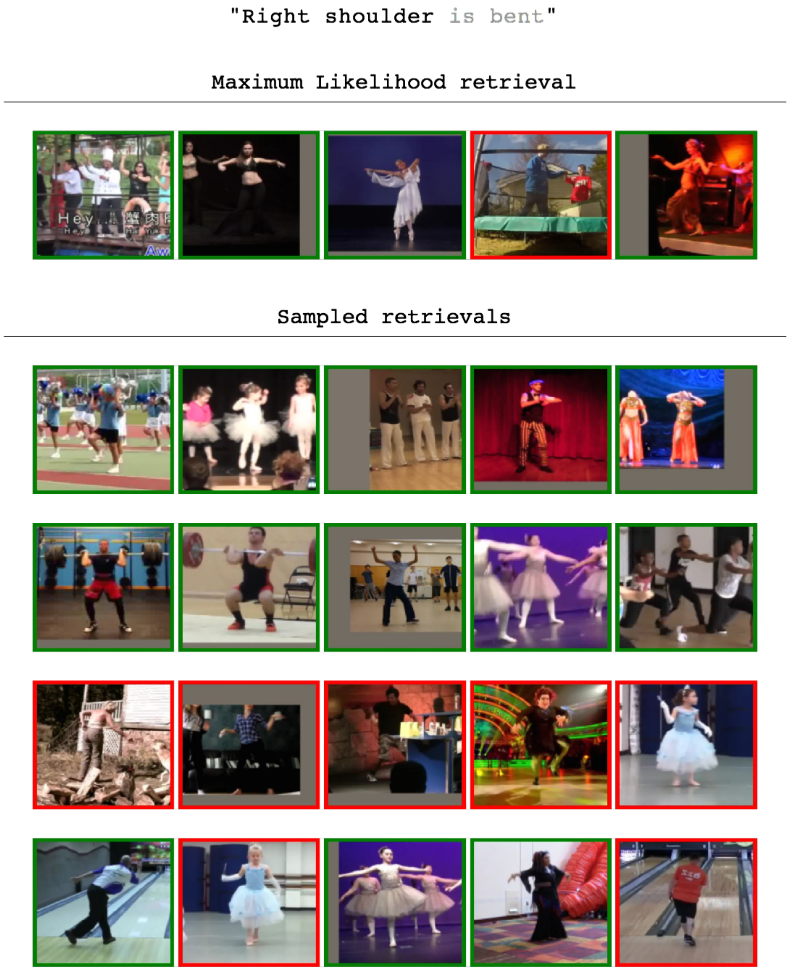
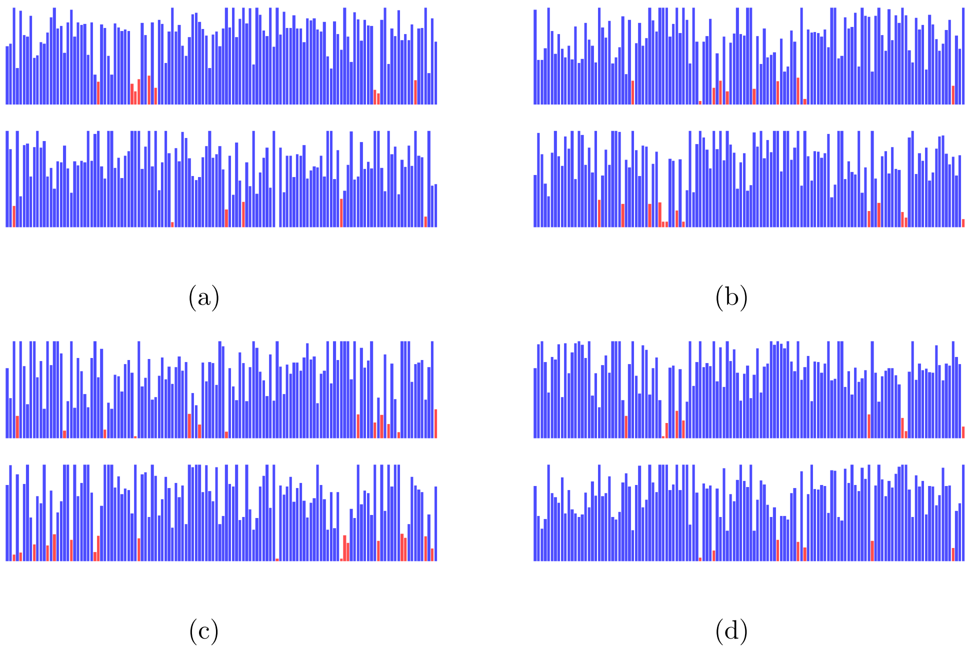


**Figure 5.15:** The first five retrievals for a Conditional Posebyte query generated using both maximum likelihood and sampling. A green and red box indicates a match and error, respectively. Top: The query results of a Posebyte completed using Maximum Likelihood, given the prior "right shoulder is bent." Bottom: Four queries generated by sampling the conditional Gaussian distribution, given the same priors, and the first five retrieval of each query, organized by row. Maximum likelihood produces Posebytes well suited for semantically similar retrieval, but is limited to retrieving entries surrounding a single location on the embedding surface, resulting in holistically similar images. On the other hand, an aggregate of sampled Posebytes provide increased visual variety, but at the expense of retrieval quality.

**Query-Aware Masks** Querying the original holistically defined embedding space presents the limitation that any one subset of conditions will not lie at a singular location. For example, while “left knee is bent” is true for someone curled up in a ball to avoid the aggression of an angry bear, or someone standing mostly upright while ascending a set of stairs, these two pose embedding vectors will be found at significantly different locations on the embedding surface. As opposed to Conditional Posebytes which attempt to deduce input examples which embed to locations optimal for a conditional match, Query-Aware Masks instead propose a simple point-wise warping to alter the embedding space such that the entries-of-interest lay closest to the origin.

Figure 5.14 (top) outlines a comparison between Query-Aware Masks, Conditional Posebytes and querying with the validation set (Mean Query) Posebytes. Much like Conditional Posebytes, Query-Aware Masks outperform a Mean Query by 13% to 21%, and display an improvement over Conditional Posebytes by 2% to 6%. While Conditional Posebytes improve upon the Mean Query by identifying optimal bit configurations for conditional matching in the embedding space, Query-Aware Masks top Conditional Posebytes by avoiding the holistically defined embedding space problem entirely. As illustrated in Figure 5.14 (bottom), two-question Query-Aware Masks maintain a large improvement over both two-question Conditional Posebytes and a two-question Mean Query.

The efficacy of Query-Aware Masks speaks to the natural tendency of pose-aware embedding networks (and possibly neural networks as a whole) to inherently factorize the embedding space, without explicit enforcement. Example masks are illustrated in Figure 5.16, demonstrating that the positive and negative state of each query subset lives on a mutually exclusive, and possibly



(a)

(b)

(c)

(d)

**Figure 5.16:** The Query-Aware Mask model learns a set of point-wise multiplications, for the purpose of warping the embedding space such that the desired entries lay closest to the origin. The four examples of learned Query-Aware Masks included here are (a) “is right shoulder bent?,” (b) “is left wrist near head,” (c) “is left knee near right knee?,”and (d) “is left wrist beyond pelvis?,” with the top and bottom rows indicating the “no” and “yes” condition, respectively. Each mask dimension with a scalar of less than 0.3 is marked in red, to indicate the approximate subspace of which the particular condition resides on. Note that the mask dimensions between the “no” and “yes”” conditions appear to be almost entirely mutually exclusive. The model was never trained to explicitly linearly factorize these concepts, and thus this observation implies that the network implicitly learns to perform something akin to a linear separation.

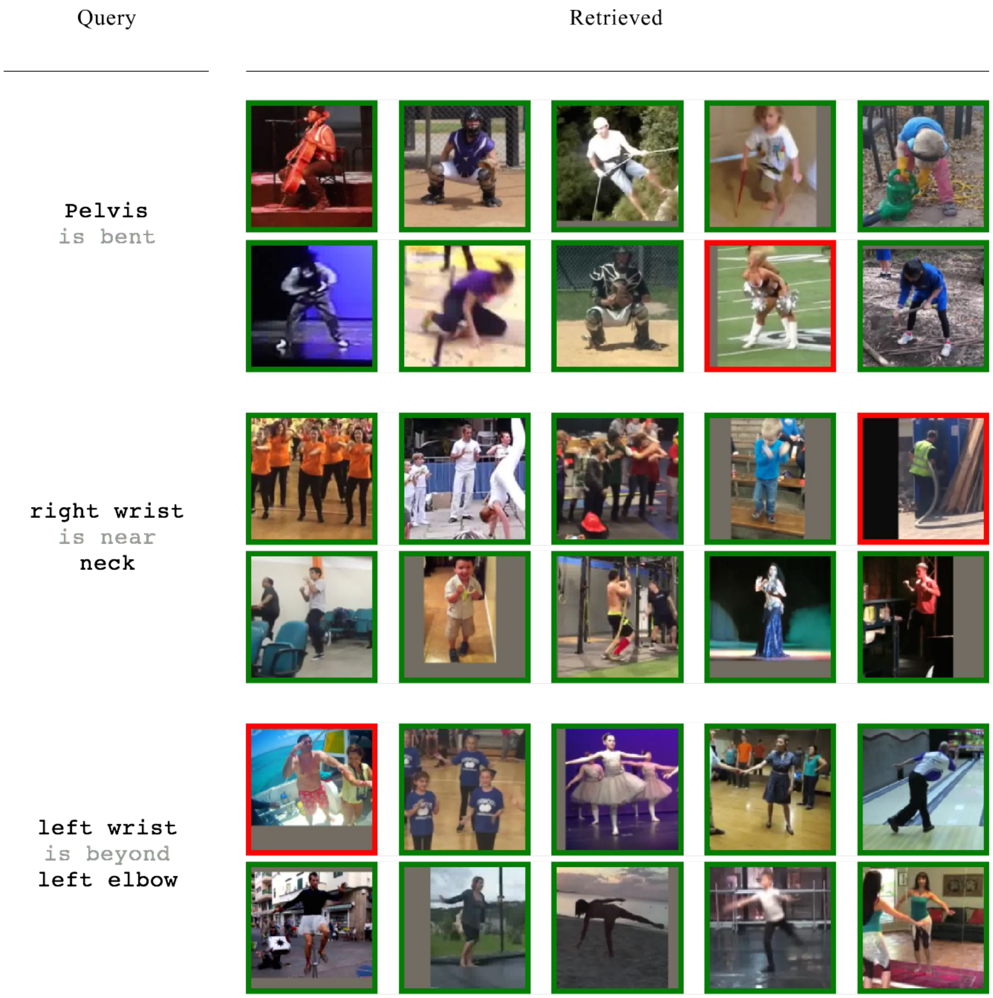


**Figure 5.17:** The results of querying with the three types of language-primitives: joint angle, joint distance, and joint relative-distance. The top and bottom row of the retrieved images presents queries of Conditional Posebytes and Query Masks, respectively. Some error cases are presented here, indicated in red, and are caused by (from top to bottom) a joint angle near the threshold, occlusion, and foreshortening. While it could be argued that perceptually these three images meet the question criteria, the fact that a hard threshold must be defined, paired with the noise of the underlying data, the occasional false negative is bound to occur.

linear, sub-space.

Example single-question queries for both Conditional Posebytes (top row) and Query-Aware Masks (bottom row) are presented in Figure 5.17, with both success and failure cases.

Surprisingly, Query-Aware Masks do not arbitrarily order the dataset embedding vector entries. While it should be possible for the model to place embedding vectors on either side of some implicitly defined threshold, the network instead appears to be sorting each entry by its degree of visual saliency. Some example queries and their respective uniformly sampled entries are presented in Figure 5.18. One possible reason for this emergent phenomenon is that the extreme cases, of any one state, are far less likely to be confused, as opposed to those which reside closer to the defining threshold. For example, if the threshold for "left elbow is bent" is defined as any angle greater than 115° then one should expect that an elbow bent at 180° is more visually apparent and therefore easier to correctly identify than an elbow bent at 116°.

## 5.4 Discussion

Pose-aware embedding networks have demonstrated their ability to inherently learn pose, without an explicit joint localization signal. Their ability to accept queries of either image or language for semantic similarity retrieval, and their tendency to factorize pose-related concepts, speaks to the capacity of these models to successfully disentangle pose information.

The proposed models, in a semi-supervised fashion, learn to disentangle pose features from pose-invariant features, such as background, foreground objects, articles of clothing, and gender. Further, between PoseEmb, PoseEmb-3D, and PoseEmb-Pro and PoseEmb-Pb, this architecture also demonstrates

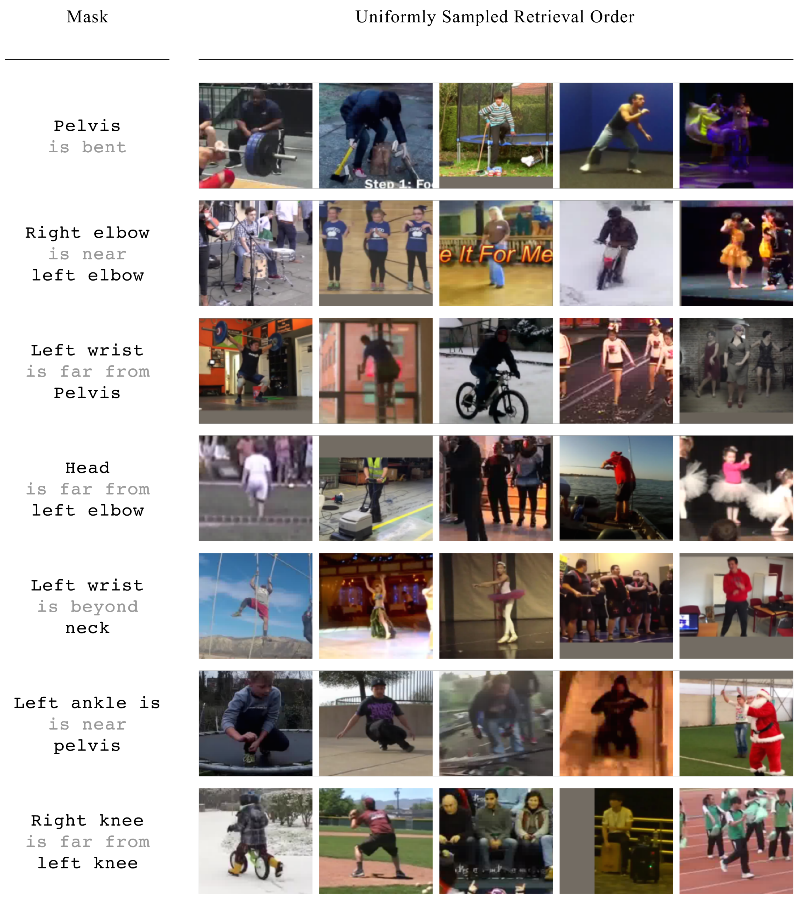


**Figure 5.18:** Query-Aware Masks appear to be taking advantage of the degree of visual saliency present within each image. The embedding space is warped using the appropriate mask and the embedding vectors are ordered by their distance from the origin. For visualization purposes (avoiding the strong bias of certain bits) the first three presented images, in each row, are uniformly sampled from the ordered embedding vectors within the count of matching ground-truth entries, and the latter two are uniformly sampled from the remaining sorted embeddings. That is, given a 1:3 positive-negative ratio from a 1,000 sorted embedding vector dataset, three are uniformly selected from the first 250, and two are uniformly selected from the remaining entries.

the ability to learn pose which is completely invariant to camera position and orientation.

Finally, as demonstrated by Query-Aware Masks, pose-aware embedding networks have an inherent tendency to factorize pose into smaller pose-related components. Recall, the pose-aware embedding space is defined holistically on a language-primitive pose descriptor with no enforcement of explicit factorization. Yet, pose-aware embedding networks must be learning something akin to a linear factorization, otherwise Query-Aware Masks, a simple linear operation, would not work.

Given the stated evidence, this evaluation concludes that pose-aware embedding networks do indeed learn to represent human pose.

# Chapter 6

# Conclusion

## 6.1 Thesis Summary

This thesis explored a family of pose-aware embedding networks defined over a variety of metrics, providing the following four contributions.

First, pose-aware embedding networks have demonstrated their capacity to disentangle human pose from raw pixel data. These models learn to be invariant to background scenery, foreground objects, articles of clothing, and gender, and with the correct metric can even learn to be invariant to camera position and orientation. Further, the camera invariant Procrustes similarity model demonstrates that removing the inherent camera perspective allows the model to retrieve entries with pose descriptors more similar to the query, by a modest 3mm on average.

Second, a multi-modal image-language embedding space was presented, providing a mechanism for a more natural means of conducting queries. That is, one may want to query for images using language, and thus with a language-primitive pose descriptor query, semantically relevant images may be retrieved.

Providing this additional language mode allows for a query approach which outperforms its image counterpart, under the hit@k metric, by 3% to 14%.

Third, two approaches for circumventing the limitations of holistically defined embedding spaces are presented: Conditional Posebytes and Query-Aware Masks. Given that pose-aware embedding networks are learned using a metric defined over the entirety of the pose descriptor, the learned embedding space is definitionally holistic. That is, any query will return results which are maximally similar to the query, overall. These two approaches circumvent this problem from both the data-end and embedding-end of the model, by generating maximally likely language-primitive queries, and by learning a linear subspace warping such that the desired entries lay closest to the origin, respectively. Conditional Posebytes and Query-Aware Masks improve recall by 13% and 17%, respectively, when compared to querying with existing Posebytes.

Fourth, a dataset specifically designed for training and evaluating pose-aware embedding networks was constructed, featuring over 24,000 2D joint, 3D joint, and language-primitive annotated images.

These contributions were implemented, using the PyTorch [71] neural network framework.

## 6.2 Future Work

There are three clear directions in which this work can be taken. First, extending the exploration of metric spaces to include video. Second, exploring potential video based language-primitives, possibly as an extension to Posebytes [72]. Third, determining if the tendency to linearly factorize relevant features is a universal phenomenon, inherent to all similarity embedding spaces.

The proposed embedding networks were defined over a mean per-joint distance and language-primitive Hamming distance. These metrics can easily be extended into the spatiotemporal domain by simply averaging across the entirety of provided frames. More so, poses in motion may be better understood by considering a distance metric which accounts for joint dynamics. An early exploration of video based pose embedding networks was conducted, however time was a limiting factor. Thus, while the dataset proposed in this thesis does provide a set of video clips from which video embedding networks may potentially be trained, this thesis refrained from moving beyond images.

This thesis presented a multi-modal image-language network, capable of retrieving images for queries of both images and language. The language was defined as Posebyte [72] language-primitive descriptors, cataloging the inter-joint relationship of an actor in an image. The logical next step would be to expand the language set to account for human pose dynamics, a study which has received little traction. Two approaches have previously been explored, with [51] defining a set of language-primitives, some of which concisely describe motion for an entire video clip, such as "torso twist," and [63], a set of primitive-language descriptors remarkably similar to Posebytes [72], each of which describes a part of an actor's pose for a single still image. One possible approach would be to consider the conjunction of joint-states. For example, the spatiotemporal counterpart of "left elbow is (not) bent" could be "left elbow bends," "left elbow straightens," "left elbow stays bent," and "left elbow stays straight."

This thesis observed that pose-aware embedding networks inherently learn to linearly factorize pose-related concepts, without explicit enforcement. Query-Aware Masks presented an approach, strongly inspired by conditional similarity networks [88] and prototype learning [30], which collapses the relevant embedding subspace such that the entries of interest lay closest to the origin.

This was achieved using a simple linear transformation, implying that pose-aware embedding networks not only learn to distinguish between pose-related concepts, but also neatly factorize them, without explicit enforcement. One obvious direction of exploration would be to determine if this emergent property is universal. If it is the case that embedding spaces inherently learn to linearly factorize the concepts which define their latent metric similarity, then a “Query-Aware Masks”-style prototype learning approach may be applied beyond pose-aware embedding networks.

# References

[1] H. Altwaijry, E. Trulls, J. Hays, P. Fua, and S. Belongie. Learning to match aerial images with deep attentive architectures. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3539–3547. IEEE, 2016.

[2] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3319–3327. IEEE, 2017.

[3] Y. Bengio, N. Boulanger-Lewandowski, and R. Pascanu. Advances in optimizing recurrent networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8624–8628. IEEE, 2013.

[4] Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. In *International Conference on Machine Learning*, pages 41–48. ACM, 2009.

[5] A. Bilal, A. Jourabloo, M. Ye, X. Liu, and L. Ren. Do convolutional neural networks learn class hierarchy? *IEEE Transactions on Visualization and Computer Graphics*, 24(1):152–162, 2018.

[6] L. Bourdev, S. Maji, T. Brox, and J. Malik. Detecting people using mutually consistent poselet activations. In *European Conference on Computer Vision*, pages 168–181. Springer, 2010.

[7] L. Bourdev, S. Maji, and J. Malik. Describing people: A poselet-based approach to attribute classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1543–1550. IEEE, 2011.

[8] L. Bourdev and J. Malik. Poselets: Body part detectors trained using 3d human pose annotations. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1365–1372. IEEE, 2009.

[9] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[10] L. Castrejon, Y. Aytar, C. Vondrick, H. Pirsiavash, and A. Torralba. Learning aligned cross-modal representations from weakly aligned data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2940–2949, 2016.

[11] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *Proceedings of the British Machine Vision Conference*, 2014.

[12] G. Chechik, V. Sharma, U. Shalit, and S. Bengio. Large scale online learning of image similarity through ranking. *Journal of Machine Learning Research*, 11(Mar):1109–1135, 2010.

[13] A. Choromanska, M. Henaff, M. Mathieu, G. B. Arous, and Y. LeCun. The loss surfaces of multilayer networks. In *Artificial Intelligence and Statistics*, pages 192–204, 2015.

[14] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[15] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 886–893. IEEE, 2005.

[16] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, pages 1–38, 1977.

[17] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[18] K. G. Derpanis, M. Sizintsev, K. J. Cannons, and R. P. Wildes. Action spotting and recognition based on a spatiotemporal orientation analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(3):527–540, 2013.

[19] A. Dosovitskiy and T. Brox. Inverting visual representations with convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4829–4837, 2016.

[20] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2758–2766, 2015.

[21] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

[22] A. A. Efros, A. C. Berg, G. Mori, and J. Malik. Recognizing action at a distance. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 726–733, Nice, France, 2003.

[23] J. L. Elman. Learning and development in neural networks: The importance of starting small. *Cognition*, 48(1):71–99, 1993.

[24] D. Erhan, Y. Bengio, A. Courville, and P. Vincent. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341(3):1, 2009.

[25] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, June 2010.

[26] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing objects by their attributes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1778–1785. IEEE, 2009.

[27] R. Fong and A. Vedaldi. Net2vec: Quantifying and explaining how concepts are encoded by filters in deep neural networks. *arXiv preprint arXiv:1801.03454*, 2018.

[28] R. C. Fong and A. Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3449–3457, 2017.

[29] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, T. Mikolov, et al. Devise: A deep visual-semantic embedding model. In *Advances in Neural Information Processing Systems*, pages 2121–2129, 2013.

[30] Y. Fu and L. Sigal. Semi-supervised vocabulary-informed learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5337–5346, 2016.

[31] G. B. Goh, N. O. Hodas, and A. Vishnu. Deep learning for computational chemistry. *Journal of computational chemistry*, 38(16):1291–1307, 2017.

[32] A. Gordo, J. Almazán, J. Revaud, and D. Larlus. Deep image retrieval: Learning global representations for image search. In *European Conference on Computer Vision*, pages 241–257. Springer, 2016.

[33] K. Hara, H. Kataoka, and Y. Satoh. Learning spatio-temporal features with 3d residual networks for action recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2, page 4, 2017.

[34] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[35] S. Hochreiter, Y. Bengio, P. Frasconi, J. Schmidhuber, et al. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies, 2001.

[36] S. J. Hwang and L. Sigal. A unified semantic embedding: Relating taxonomies and attributes. In *Advances in Neural Information Processing Systems*, pages 271–279, 2014.

[37] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339, 2014.

[38] H. Jhuang, T. Serre, L. Wolf, and T. Poggio. A biologically inspired system for action recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1–8. Ieee, 2007.

[39] M. Kahng, P. Y. Andrews, A. Kalro, and D. H. P. Chau. Activis: Visual exploration of industry-scale deep neural network models. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):88–97, 2018.

[40] A. Karpathy and L. 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.

[41] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

[42] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[43] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

[44] S. Kwak, M. Cho, and I. Laptev. Thin-slicing for pose: Learning to understand pose without explicit pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4938–4947. IEEE, 2016.

[45] C. H. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 951–958. IEEE, 2009.

[46] Y. Le Cun. Learning process in an asymmetric threshold network. In *Disordered Systems and Biological Organization*, pages 233–240. Springer, 1986.

[47] L. Leal-Taixé, A. Milan, I. Reid, S. Roth, and K. Schindler. Motchallenge 2015: Towards a benchmark for multi-target tracking. *arXiv preprint arXiv:1504.01942*, 2015.

[48] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[49] S. Li and A. B. Chan. 3d human pose estimation from monocular images with deep convolutional neural network. In *Asian Conference on Computer Vision*, pages 332–347. Springer, 2014.

[50] S. Li, W. Zhang, and A. B. Chan. Maximum-margin structured learning with deep networks for 3d human pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2848–2856, 2015.

[51] J. Liu, B. Kuipers, and S. Savarese. Recognizing human actions by attributes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3337–3344. IEEE, 2011.

[52] P.-L. Loh and M. J. Wainwright. High-dimensional regression with noisy and missing data: Provable guarantees with nonconvexity. In *Advances in Neural Information Processing Systems*, pages 2726–2734, 2011.

[53] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[54] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[55] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[56] A. Mahendran and A. Vedaldi. Understanding deep image representations by inverting them. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5188–5196, 2015.

[57] S. Maji, L. D. Bourdev, and J. Malik. Action recognition from a distributed representation of pose and appearance. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3177–3184, 2011.

[58] K. Matzen, K. Bala, and N. Snavely. Streetstyle: Exploring world-wide clothing styles from millions of photos. *arXiv preprint arXiv:1706.01869*, 2017.

[59] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[60] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119, 2013.

[61] G. A. Miller. Wordnet: A lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[62] G. Mori, C. Pantofaru, N. Kothari, T. Leung, G. Toderici, A. Toshev, and W. Yang. Pose embeddings: A deep architecture for learning to match human poses. *arXiv preprint arXiv:1507.00302*, 2015.

[63] M. Müller, T. Röder, and M. Clausen. Efficient content-based retrieval of motion capture data. In *ACM Transactions on Graphics*, volume 24, pages 677–685. ACM, 2005.

[64] A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. In *European Conference on Computer Vision*, pages 483–499. Springer, 2016.

[65] A. Nguyen, J. Yosinski, and J. Clune. Multifaceted feature visualization: Uncovering the different types of features learned by each neuron in deep neural networks. *Visualization for Deep Learning Workshop, International Conference in Machine Learning*, 2016. arXiv preprint arXiv:1602.03616.

[66] Q. Nguyen and M. Hein. The loss surface of deep and wide neural networks. *arXiv preprint arXiv:1704.08045*, 2017.

[67] Q. Nguyen and M. Hein. Optimization landscape and expressivity of deep cnns. In *International Conference on Machine Learning*, pages 3727–3736, 2018.

[68] M. Norouzi, T. Mikolov, S. Bengio, Y. Singer, J. Shlens, A. Frome, G. Corrado, and J. Dean. Zero-shot learning by convex combination of semantic embeddings. In *International Conference on Learning Representations*, 2014.

[69] O. M. Parkhi, A. Vedaldi, A. Zisserman, et al. Deep face recognition. In *Proceedings of the British Machine Vision Conference*, volume 1, page 6, 2015.

[70] R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, pages 1310–1318, 2013.

[71] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

[72] G. Pons-Moll, D. J. Fleet, and B. Rosenhahn. Posebits for monocular human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2337–2344, 2014.

[73] F. Rosenblatt. *The Perceptron, A Perceiving And Recognizing Automaton (Project Para).* Cornell Aeronautical Laboratory, 1957.

[74] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533, 1986.

[75] J. Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

[76] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 815–823, 2015.

[77] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 618–626, 2017.

[78] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[79] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in Neural Information Processing Systems*, pages 568–576, 2014.

[80] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[81] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

[82] S. Streuber, M. A. Quiros-Ramirez, M. Q. Hill, C. A. Hahn, S. Zuffi, A. O'Toole, and M. J. Black. Body talk: Crowdshaping realistic 3d avatars with words. *ACM Transactions on Graphics*, 35(4):54, 2016.

[83] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015.

[84] T. Tieleman and G. Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.

[85] D. Tome, C. Russell, and L. Agapito. Lifting from the deep: Convolutional 3d pose estimation from a single image. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2500–2509, 2017.

[86] A. Toshev and C. Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1653–1660, 2014.

[87] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4489–4497. IEEE, 2015.

[88] A. Veit, S. Belongie, and T. Karaletsos. Conditional similarity networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[89] I. Vendrov, R. Kiros, S. Fidler, and R. Urtasun. Order-embeddings of images and language. *arXiv preprint arXiv:1511.06361*, 2015.

[90] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3156–3164, 2015.

[91] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu. Learning fine-grained image similarity with deep ranking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1386–1393, 2014.

[92] L. Wang, Y. Qiao, and X. Tang. Action recognition with trajectory-pooled deep-convolutional descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4305–4314, 2015.

[93] L. Wang, Y. Xiong, Z. Wang, and Y. Qiao. Towards good practices for very deep two-stream convnets. *arXiv preprint arXiv:1507.02159*, 2015.

[94] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *European Conference on Computer Vision*, pages 20–36. Springer, 2016.

[95] Y. Wang and G. Mori. A discriminative latent model of object classes and attributes. In *European Conference on Computer Vision*, pages 155–168. Springer, 2010.

[96] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh. Convolutional pose machines. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4724–4732, 2016.

[97] P. J. Werbos. Applications of advances in nonlinear sensitivity analysis. In *System Modeling and Optimization*, pages 762–770. Springer, 1982.

[98] H. Wu, M. Merler, R. Uceda-Sosa, and J. R. Smith. Learning to make better mistakes: Semantics-aware visual food recognition. In *Proceedings of the ACM Conference on Multimedia*, pages 172–176. ACM, 2016.

[99] S. Xie and Z. Tu. Holistically-nested edge detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1395–1403, 2015.

[100] 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 *International Conference on Machine Learning*, pages 2048–2057, 2015.

[101] B. Yao, X. Jiang, A. Khosla, A. L. Lin, L. Guibas, and L. Fei-Fei. Human action recognition by learning bases of action attributes and parts. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1331–1338. IEEE, 2011.

[102] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson. Understanding neural networks through deep visualization. In *Deep Learning Workshop, International Conference on Machine Learning*, 2015.

[103] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision*, pages 818–833. Springer, 2014.

[104] W. Zhang, M. Zhu, and K. G. Derpanis. From actemes to action: A strongly-supervised representation for detailed action understanding. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2248–2255, 2013.

[105] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2921–2929. IEEE, 2016.

[106] X. Zhou, M. Zhu, S. Leonardos, K. G. Derpanis, and K. Daniilidis. Sparseness meets deepness: 3d human pose estimation from monocular video. In *Proceed-*

*ings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4966–4975, 2016.