Distance Metrics for Clustering in Regards to Hypergraph-Based Human Pose Image Retrieval Systems

Juan Diego Castro
Computer Science UTEC

Kevin Huaman
Computer Science UTEC

Juan Diego Prochazka
Computer Science UTEC

Abstract

Hypergraph retrieval systems like [20] require the computation of cluster centroids in the indexed data as part of the creation of the structure. In this paper, we explore a variety of distance functions to determine which yields the better result both in terms of efficiency and effectiveness in action similarity search over poses when using the referenced retrieval system. We introduce new Joint-Angle and Composite distances which grant better performance than traditional alternatives, and which easily generalize to other keypoint based data representations.

1 Introduction

When building the hypergraph based system described in [20], there arises the need for the construction of clusters in the input data. These clusters define the hypergraph [3], which allows efficient similarity search in a database. Hence, the method through which one should arrive at such centroids bears a large importance in both the quality and speed of query responses. In this paper we explore distance functions for the application of this system to the indexing of human pose.

Determining clusters within data when the number of clusters is unknown (as required by the studied algorithm) is a well studied problem in the field of unsupervised machine learning, and several methods have been described in the literature for this purpose [6] [13]. However, the specific system we are studying calls for a modified Leader's [16] algorithm described in the same paper, which adds desirable properties such as stability and a guarantee of separability. Given a clustering method, the idea of changing the way distance is measured between objects is not new, and different distance functions have been used extensively in this area [22] [23].

Figure 1: Hypergraph Construction example

Human pose similarity has has been previously addressed via pose skeleton interpolation from an input image, such as those detailed in [5] [18], [9] and [8] and then building an index over the extracted

Final Paper for the ML3061-2023.2 - Machine Learning Class at UTEC. Lima, Peru.

features [21]. More recent approaches approach the problem via direct embedding extraction, such as in [24] [17] [14]. The first category is more mature, some of the methods presented have off-the-shelf library implementations available [2] and will be the focus of our study.

The poses we will be using consist of a collection of keypoints (known points) in a human body existing in 3D space provided by the method described in [2]. Because the same keypoints exist in all poses, there exist two interpretations of the data. The first one is thinking about each keypoint individually, and defining the distance as a function of other distances over the points. The other interpretation corresponds to thinking about the entire collection of keypoints as components of a single vector.

We compare well known Manhattan and Euclidean distance metrics with our own Joint-Angle and Composite distances in regards to their performance in Hypergraph based Image Retrieval. The comparison criteria include query time, precision and recall over the retrieved data.

1.1 Contributions

Our contribution lies in the proposal of two new distance functions for human pose and the comparison of their effectiveness against known functions in the context of hypergraph based human pose image retrieval systems. Knowing the strengths and weaknesses of distance functions for this application allows the construction of an efficient indexing system that does not rely in computationally expensive neural networks as much (both during a query and when building the index), and that does not require any training.

2 Our Approach

2.1 Hypergraph

A hypergraph is a generalization of a graph where edges can connect more than 2 vertices (these are called hyperedges). The hypergraph is defined [3] as follows:

Let $H = (\vartheta, \xi)$ be a hypergraph where $\vartheta = \{v_1, v_2, \ldots, v_n\}$ is a finite set of vertices and $\xi = \{e_1, e_2, \ldots, e_m\}$ is a family of subsets of ϑ, known as hyperedges. This definition is subject to the restriction that $\forall e \in \xi, e \neq \emptyset$ and $\bigcup_{e \in \xi} e = \vartheta$. The first condition interprets the existence of a hyperedge in H as requiring it to contain at least one element of ϑ. On the other hand, the second restriction implies that there can not exist a vertex in ϑ such that it does not belong to one hyperedge at least.

For example, consider the hypergraph shown at the end of the process in figure 1. Then $\vartheta = \{a, b, c, d, e, f, g, h, i\}$ and $\xi = \{\{a, b, c, d, e, f\}, \{d, g, h\}, \{f, i, j\}\}$

The hypergraph has been used in the pattern recognition domain, for object representation [7], similarity measures [19], object clustering [10] and category retrieval [20]. Specifically, category retrieval is a paradigm that consists of retrieving the maximum amount of images that belongs to a specific class, and we aim to use the hypergraph structure to perform such retrieval method.

2.2 Hypergraph-based model

In our dataset, each image contains a full body person carrying out some activity on an specific pose. The first task to perform is to extract the features from the image and obtain a characteristic graph for each pose in the dataset. Our hypergraph model uses these graphs as vertices. Then, each hyperedge in H is a cluster that groups the poses. In the context of unsupervised learning and clustering it is of utmost importance for our hypergraph to maintain similar poses in the same hyperdege, because then a hyperedge could be associated not only to a subset of ϑ, but to a determined class that groups an activity or intention on the human poses. Keep in mind that in this study we will consider the terms hyperedge, cluster and cluster centroid to be somewhat equivalent, as the role that these play are almost identical.

Note that as there is no restriction about the elements contained on any subset in ξ, more than they must belong to ϑ, then the model admits overlapping between clusters. This makes possible that if a pose is related to more than one activity, the algorithm studied in [20] should add the object on all

of those related clusters. The added value of this structure in comparison with a classical clustering method lies in the possibility of assign a object to more than one class. This aims to add flexibility to the model during inference by allowing the retrieval of the same image in response of two different queries that, in principle, sought their responses in different clusters. This approach lets us to perform category retrieval queries by looking for a cluster $e \in \xi$ such that the elements contained in e are the most similar as possible as the query graph and return those elements. A construction over the definition of a hypergraph uses the concept of centroid of a hyperedge to perform such task.

Given a set of graphs S, the median graph [26] $\hat{g}$ of S is defined as:

$$\hat{g} = \arg\min_{g \in S} \sum_{i=1}^{|S|} d\left(g, g_i\right), \text{where } d \text{ is a distance function.}$$

Let $H = (\vartheta, \xi)$ be a hypergraph where ϑ is a set of graphs. Then, the centroid of each $e \in \xi$ is $\hat{g}(e)$. Now the category retrieval task is reduce to compute all the distances between the query graph and the clusters centroids and return the elements that belongs to the cluster with the nearest centroid, as illustrated in Algorithm 1.

Given this context, the importance of finding the best distance functions between graphs becomes evident as it is crucial to determine the most effective method to build the clusters and to perform queries, as we can see in Figure 2.

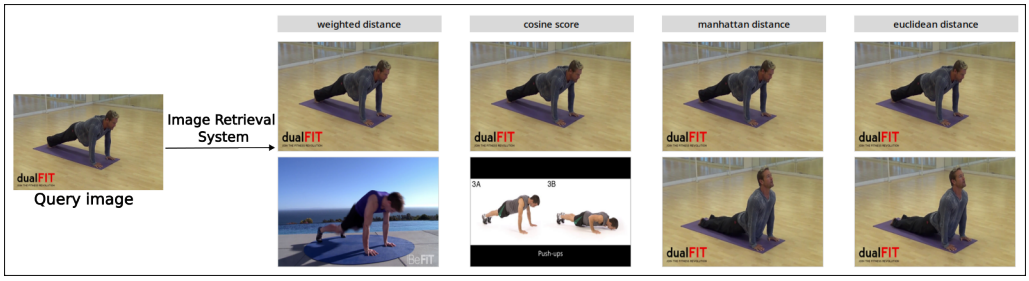

Figure 2: Query responses varying the distance function

2.3 Distances

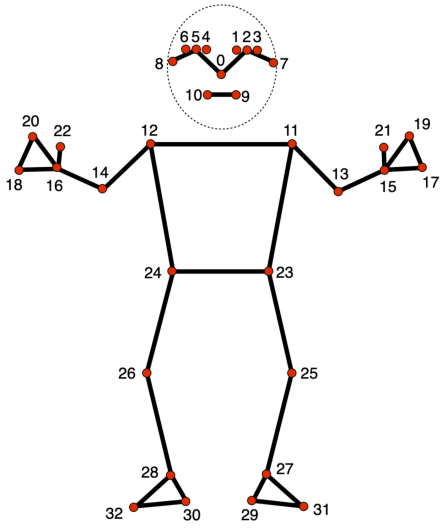

Figure 3: Pose Keypoints, taken from [12]

For our purposes, a pose is a graph G with 33 nodes (x, y, z) and edges as seen in Figure 3 (note that the figure indexes from zero). Because the graphs all have the same vertices and edges, we define two new interpretations for the data. The first one is a list of keypoints L with $|L| = 33$ where $v_i = (x_i, y_i, z_i) \in L$. The other one is a flat vector V, where $|V| = 99$ composed of the concatenation of all the keypoints in ascending order. The native graph, vector list and vector interpretations of our data inspired the creation or usage of the distance functions described below:

Manhattan Distance We will use the classic version of Manhattan distance $d_m(V_1, V_2)$ over the flat vector interpretation. Specifically, let a_i and b_i be the i'th element of V_1 and V_2 respectivelly, then:

$$d_m(V_1, V_2) = \sum_{i=1}^{99} |a_i - b_i|$$

Euclidean Distance Traditional euclidean distance computes the square root of the sum of the differences on the coordinates of pairs of objects. We build on this distance by defining Vector List Euclidean Distance as the sum of all euclidean distances over $j_i = (x_{ji}, y_{ji}, z_{ji}) \in L_1, k_i = (x_{ki}, y_{ki}, z_{ki}) \in L_2$ where k_i, j_i are the i'th vector in their corresponding list. More precisely, we define the euclidean distance between vector lists as

$$d_e(L_1, L_2) = \sum_{i=0}^{32} \sqrt{(x_{ji} - x_{ki})^2 + (y_{ji} - y_{ki})^2 + (z_{ji} - z_{ki})^2}$$

We use this distance as it is the more natural space for points in 3D space, and should place similar poses nearby.

Joint-Angle Distance We define the Joint-Angle or cosine distance for human poses as the sum of the cosine distances between corresponding edges of the graph. More formally, let G_1, G_2 be graphs representing pose. Let $E(G)$ be the list of edges of a pose graph, then each edge $(u, v) \in E(G)$ can be though of as a vector $\vec{uv}$ starting in the keypoint u and finishing at v. Note that different poses G_1 and G_2 will have one such vector per edge in $E(G)$ denoted as $\vec{uv}_1$ and $\vec{uv}_2$ respectively.Then, the Joint-Angle distance $d_a(G_1, G_2)$ is computed as

$$d_a(G_1, G_2) = \sum_{(u,v) \in E(G)} \left\{ 1 - \frac{\vec{uv}_1 \cdot \vec{uv}_2}{|\vec{uv}_1||\vec{uv}_2|} \right\}$$

We believe this distance function will be invariant to translation, a property desired in pose similarity.

Composite Distance We define a composite distance over L_1, L_2 as the sum of the Joint-Angle distance and the square root of Euclidean vector list distance, in precise terms let $d_c(L_1, L_2)$ denote the composite distance between L_1 and L_2. Let $k \in \mathbb{R}^+$ be a weight factor then:

$$d_c(L_1, L_2) = d_a(L_1, L_2) + k \cdot d_e(L_1, L_2)$$

We believe that this distance function has the possibility of combining the invariance of cosine to translation with the naturallity of euclidean distance for this dataset.

2.4 System Parameters

For our composite distance we use $k = 0.5$, for the threshold required by the clustering method we use

3 Experimentation

We will be using the MPII dataset [1], which contains human images with associated activity descriptors. We generated pose estimations using the MediaPipe library for python [4]. The library returns a graph of keypoints with data consisting of 4D vectors, where the first three dimensions encode a position in 3D space, and the fourth parameter encodes visibility in a normalized 0 to 1 range where 1 is fully visible. We filtered the dataset so that only images with an average visibility

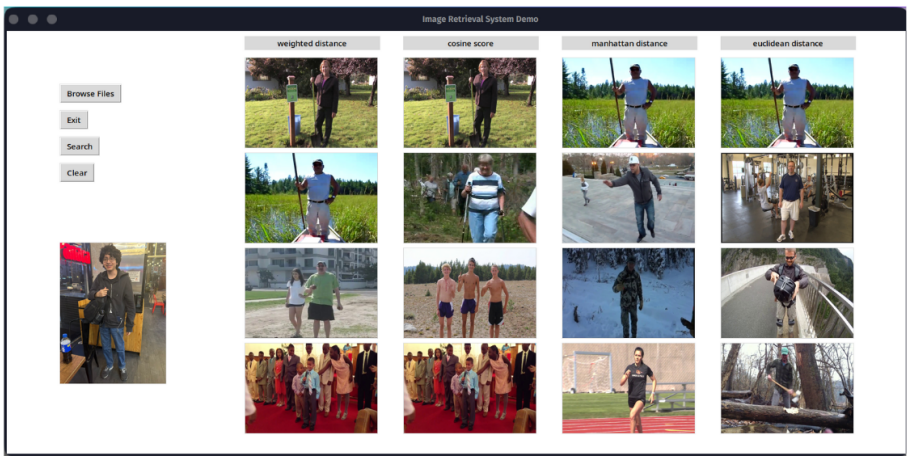

Figure 4: Query results for an image out of the dataset

over all the keypoints is higher than or equal to 0.8 remain, leaving us with 6944 images. We ignore the fourth dimension for all subsequent tasks.

Our performance evaluation will consist of a query image containing a pose and tagged with an action that exists in the database by a human. To quantify the results, we will use the precision measure detailed in [25]. We do not report recall because the system is a k-nearest type, which limits the maximum possible recall to k, making this metric meaningless. Given that this is an indexing system we will also report query times.

A number of queries where made in the system consisting of human-tagged images that do not appear in the dataset. We consider a retrieved image a positive if the retrieved image has an identical activity tag as the human's. The testing data was of not more than 100 images from outside the dataset and hand-tagged by the researchers, each yielding query results such as those seen in 4 Our results are detailed in table 1. Table 2 contains metrics that give insight into the structure of the hypergraph and the hyperedges (clusters), detailed in [15], for each distance evaluated.

Table 1: Performance Evaluation Metrics for Query Retrieval

Distance Function	**Precision (%)**	**Query Time (ms)**
Manhattan	80.000	0.016
Euclidean	78.333	0.017
Cosine	79.167	0.315
Composite	88.333	0.488

Table 2: Hypergraph Measures for System Evaluation

Distance Function	**Overlapping**	**Density**	**Number of Clusters**
Manhattan	8.084	0.506	3516
Euclidean	30.440	0.235	1632
Cosine	33.120	0.236	1636
Composite	9.851	0.552	3832

4 Insights

We discuss in detail our observations over each of the distances below. We offer explanations of the results based on the structural characteristics of the resulting hypergraph for each distance metric.

High hyperedge (cluster centroid) count distances Both Manhattan and Composite distances exhibit high hyperedge counts, along with a precision of 80% or more. We believe this phenomenon arises because more hyperedges allow the structure to more finely separate images, hence allowing for better precision. Furthermore, the ability of items to belong in more than one cluster means that the existance of more centroids does not necessarily imply that there are less images per hyperedge. That said, overlap for both distances was significantly less that for the other methods.

We believe the reason why more hyperedges exist in the Manhattan and Composite distances is that both distances return a higher distance for each two given sets of ketpoints. In the case of the well studied Manhattan distance, the property of divergence to infinity with high dimensions has been proven in [11], and amply discussed as part of the curse of dimensionality. It makes sense, thus that over our 99 dimensions the metric will tend to be relatively high. In the other hand, the composite distance will evaluate higher than both cosine and euclidean because it is the sum of the never negative euclidean and joint-angle distances. With a fixed threshold parameter, this means that more clusters will be formed.

It is important to note that hyperedge counts alone is not enough to explain the effectively of a distance metric, as the precision of Joint-Angle is comparable to that of Manhattan with very dissimilar characteristics in this regard.

Low query times We found that the importance of the cluster count for query times is smaller than that of the distance calculation efficiency for queries, as can be seen by the results of Manhattan and Composite distances which share a very similar number of cluster centroids but have notably different search query times. Furthermore, we determine that in comparison to more traditional metrics, pure Joint-Angle behaves worst in this regard while only improving results marginally. The potential of Joint-Angle is greatly increased, however, by combining it into the composite distance.

The results of the Manhattan distance The behaviour of the Manhattan distance was surprising to us, as we expected it to be one of the worst scoring. This was because the interpretation of the data that suggested its use (as a flat vector) is unnatural for a graph data structure.

We believe that the reason that Manhattan distance works reasonably well is that the Manhattan distance of the flat vector representation of the data is equivalent to the sum of the Manhattan distances over all the points in a way similar to that of the euclidean distance we defined earlier. More formally given two vectors $A =< a, b, c, d, e, f.... >$ and $B =< x, y, z, n, m, w.... >$ and their asociated vector lists $A' =<< a, b, c >, < d, e, f >, ... >$, $B' =<< x, y, z >, < n, m, w >, ... >$, their Manhattan distance $\sum |A_i - B_i| = |a - x| + |b - y| + ...$ is equal to the sum of the Manhattan distances of each element of the corresponding vector lists $\sum_{A'_i \in A', B'_i \in B'} \sum_{j=1}^{3} |A'_{ij} - B'_{ij} = (|a - x| + |b - y| + |c - z|) + (|d - n| + |e - m| + |f - w|) + ... = |a - x| + |b - y|....$

Joint-Angle and Composite distances The results of the experimentation are surprising in regards to the behaviour of Joint-Angle and Composite distances. We expected Joint-Angle to be better that almost all distances because of the translational invariance property that it exhibits, yet it behaved the worst. Even more surprising, the Composite distance which uses this metric as part of its calculation scored the highest among all of them.

We believe that absolute translational invariance may not be desirable to differantiate between activities. In its place, less sensitivity to translation may be more accurate for the task at hand. In this regard, Composite distance exploits the properties of Joint-Angle better by not completely ignoring positional information.

Overfitting During the experimentation process, queries were made to the system using images already in the dataset presenting results such as in 5. We suspect that these have a better precision than those from outside the dataset in all distance metrics, suggesting a degree of overfitting to the data. However, standard overfitting correction techniques such such as varying the threshold to be larger may decrease hyperedge count wich in light of our findings may reduce the precision of query responses.

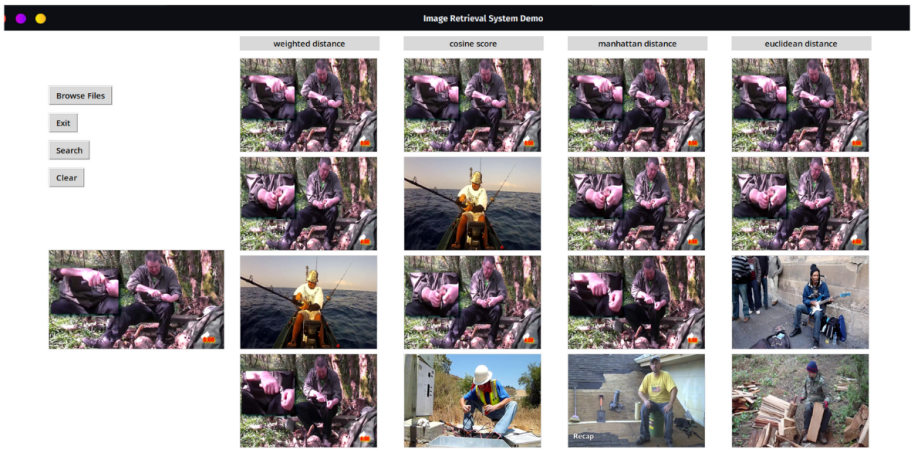

Figure 5: Query with an image from the dataset. Note than some images are very similar because MPII is created via stills from YouTube videos

5 Conclusions

We determine that our Composite distance offers the best results for this system, and that the reason for its performance is that it is not absolutely invariant, but instead has low sensitivity to translation. This allows it to consider important locational information while also acknowledging that two poses are similar regardless of where in the image they appear.

Furthermore, Manhattan distance showed considerably better query times, but a notably decreased precision. However, both Composite and Manhattan show similar density and counts of cluster centroids. We attribute this to the fact that these distance functions tend to put elements further away, allowing the clusters to contain less and better grouped elements. In contrast, the worse performing Cosine and Euclidean metrics resulted in more, highly overlapped hyperedges which would have hindered the structure's ability to retrieve images precisely.

Finally, we recognise the possibility of overfitting in our clusters, but the usual method of relaxing threshold parameters is expected to hinder the system's precision by reducing the number of hyperedges. We believe that in this regard a more thorough exploration of the threshold parameter or a larger dataset may be in order.

Due to the way our distance metrics are defined, both Joint-Angle and Composite can be used to measure similarity in applications that work with isomorphic graphs in which nodes represent points in 3d space that are connected by an edge. This 'skeleton' representation is applicable in many fields, from human pose, to buildings and molecular structure.

6 Apendix

The exact category retrieval algorithm used is shown in the next page:

Algorithm 1 Category Retrieval

procedure CATEGORYRETRIEVAL(H: hypergraph, q: graph, k: integer)
    Sort the elements $e \in H.\xi$ in non-decreasing order based on $d(\hat{g}(e), q)$
    Let $S = H.\xi[1]$, the hyperedge with the nearest centroid to q
    Let $pq = \text{MaxHeap}()$, a max heap organizing graphs by distance
    for each graph g in S **do**
        if $|pq| < k$ **then**
            pq.insert$(d(g, q), g)$ ▷ Insert graph with its distance
        else if pq.top$().d > d(g, q)$ **then**
            pq.pop()
            pq.insert$(d(g, q), g)$ ▷ Update with closer graph and its distance
        end if
    end for
    return $g \in pq$.heapsort() ▷ Return sorted graphs by distance
end procedure

Acknowledgments and Disclosure of Funding

We would like to thank UTEC Khipu cluster for granting us access to compute resources used for this investigation.

References

[1] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis:. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[2] V. Bazarevsky and I. Grishchenko. Blazepose: On-device real-time body pose tracking.

[3] C. Berge. Graphes et hypergraphes. *Paris Dunod*, 1970.

[4] J. Tang C. Lugaresi. Mediapipe: A framework for building perception pipelines.

[5] Z. Cao and G. Hidalgo. Openpose: Realtime multi-person 2d pose estimation using part affinity fields.

[6] Y. Cheng. Mean shift, mode seeking, and clustering. *IEEE Transactions on Pattern Analysis and Machine Inteligence*, 1995.

[7] Qionghai Dai and Yue Gao. *Hypergraph Modeling*. Springer Nature Singapore, 2023.

[8] L. Pishchulin E. Insafutdinov. Deepercut: A deeper, stronger, and faster multi-person pose estimation model.

[9] H.S Fang and J. Li. Alphapose: Whole-body regional multi-person pose estimation and tracking in real-time.

[10] Yue Gao, Zizhao Zhang, Haojie Lin, Xibin Zhao, Shaoyi Du, and Changqing Zou. Hypergraph learning: Methods and practices. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[11] Charu C. ggarwal, Alexander Hinneburg, and Daniel A. Keim. On the surprising behavior of distance metrics in high dimensional space. In Jan Van den Bussche and Victor Vianu, editors, *Database Theory — ICDT 2001*, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg.

[12] Google, Sep 2023. available online at `https://developers.google.com/mediapipe/solutions/vision/pose_landmarker`.

[13] Greg Hamerly and Charles Elkan. Learning the k in k-means. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems*, volume 16. MIT Press, 2003.

[14] S. Khuak and M. Cho. Thin-slicing for pose: Learning to understand pose without explicit pose estimation. *CVPR*, 2016.

[15] Geon Lee, Minyoung Choe, and Kijung Shin. How do hyperedges overlap in real-world hypergraphs? – patterns, measures, and generators, 2021.

[16] Robert F. Ling. Cluster analysis algorithms for data reduction and classification of objects. *Technometrics*, 23(4):417–418, 1981.

[17] G. Mori and C. Pantofaru. Pose embeddings: A deep architecture for learning to match human poses.

[18] G. Panpadreou and T. Zhu. Personlab: Person pose estimation and instance segmentation with a bottom-up, part-based, geometric embedding model.

[19] A. Robles-Kelly and E.R. Hancock. Graph edit distance from spectral seriation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2005.

[20] S.Tabone S. Jouily. Hypergraph-based image retrieval for graph-based representation. *Pattern Recognition*, 2012.

[21] K.S Seo S. Yang. Automatic photo indexing based on person identity.

[22] Suchita Gupta Shraddha Pandit. A comparative study on distance measuring approaches for clustering. *International Journal of Research in Computer Science*, 2011.

[23] A. Singh and A.Yadav. K-means with three different distance metrics. *International Journal of Computer Applications*, 2013.

[24] J.J Sun and J. Zhao. View-invariant probabilistic embedding for human pose.

[25] Kai Ming Ting. *Precision and Recall*, pages 781–781. Springer US, Boston, MA, 2010.

[26] X. Jianx, A. Munger, H. Bunke. On median graphs: Properties, algorithms and applications. *IEEE Transactions on Pattern Analysis and Machine Inteligence*, 2001.