A Hierarchical Tree Distance Measure for Classification

Kent Munthe Caspersen[1], Martin Bjeldbak Madsen[1], Andreas Berre Eriksen[1] and Bo Thiesson[1]
[1]*Department of Computer Science, Aalborg University, Denmark*
{swallet, martinbmadsen}@gmail.com, {andreasb, thiesson}@cs.aau.dk

Keywords: Machine Learning, Multi-class Classification, Hierarchical Classification, Machine Learning, Tree Distance Measures, Multi-output Regression, Multidimensional Scaling, Process Automation, UNSPSC.

Abstract: In this paper, we explore the problem of classification where class labels exhibit a hierarchical tree structure. Many multiclass classification algorithms assume a flat label space, where hierarchical structures are ignored. We take advantage of hierarchical structures and the interdependencies between labels. In our setting, labels are structured in a product and service hierarchy, with a focus on spend analysis. We define a novel distance measure between classes in a hierarchical label tree. This measure penalizes paths though high levels in the hierarchy. We use a known classification algorithm that aims to minimize distance between labels, given any symmetric distance measure. The approach is global in that it constructs a single classifier for an entire hierarchy by embedding hierarchical distances into a lower-dimensional space. Results show that combining our novel distance measure with the classifier induces a trade-off between accuracy and lower hierarchical distances on misclassifications. This is useful in a setting where erroneous predictions vastly change the context of a label.

1 Introduction

With the increasing advancement of technologies developed to gather and store vast quantities of data, interesting applications arise. Many kinds of business processes are supported by classifying data into one of multiple categories. In addition, as the quantity of data grows, structured organizations of assigned categories are often created to describe interdependencies between categories. Spend analysis systems are an example domain where such a hierarchical structure can be beneficial.

In a spend analysis system, one is interested in being able to drill down on the types of purchases across levels of specificity to aid in planning of procurements. Such tools also provide processes to gain insights in how much and to whom spending is going towards, supporting spend visibility. For example, in the UNSPSC[1] taxonomy, a procured computer mouse would belong to the following categories of increasing specificity: “Information Technology Broadcasting and Telecommunications”, “Computer Equipment and Accessories”, “Computer data input devices”, “Computer mouse or trackballs”. For more information on the UNSPSC standard, see (Programme, 2016).

Many classification problems have a hierarchical structure, but few multiclass classification algorithms take advantage of this fact. Traditional multiclass classification algorithms ignore any hierarchical structure, essentially flattening the hierarchy such that the classification problem is solved as a multiclass classification problem. Such problems are often solved by combining the output of multiple binary classifiers, using techniques such as One-vs-One and One-vs-Rest to provide predictions (Andersen et al., 2016; Bishop, 2006).

Hierarchical multiclass classification (HMC) algorithms are a variant of multiclass classification algorithms which take advantage of labels organized in a hierarchical structure. Depending on the label space, hierarchical structures can be in the shape of a tree or directed acyclic graph (DAG). Figure 1 shows an example of a tree-based label structure. In this paper, we focus on tree structures.

Silla and Freitas (Silla and Freitas, 2011) describe hierarchical classification problems as a 3-tuple $\langle \Upsilon, \Psi, \Phi \rangle$, where Υ is the type of graph representing the hierarchical structure of classes, Ψ specifies whether a datapoint can have multiple labels in the class hierarchy, and Φ specifies whether the labeling of datapoints only includes leaves or if nodes within the hierarchy are included as well. Using this definition, we are concerned with problems of the form

[1] United Nations Standard Products and Services Code®, a cross-industry taxonomy for product and service classification.

- $\Upsilon = T$ (tree), meaning classes are organized in a tree structure.
- Ψ = SPL (single path of labels), meaning the problems we consider are not hierarchically multilabel.
- $\Phi = PD$ (partial depth) labeling, meaning datapoints do not always have a leaf class.

In this paper, a novel distance measure is introduced, with respect to the label tree. The purpose of the distance measure is to capture similarity between labels and penalize errors at high levels in the hierarchy, more than errors at lower levels. This distance measure leads to a trade-off between accuracy and the distance of misclassifications. Intuitively, this trade-off makes sense for UNSPSC codes as, for example, classifying an apple as a fresh fruit should be penalized less than classifying an apple as toxic waste. Training a classifier for such distance measures is not straightforward, therefore a classification method is presented, which copes with a distance measure defined between two labels.

The rest of this paper is structured as follows. Section 2 discusses existing HMC approaches in the literature. Section 3 introduces hierarchical classification. In Section 4, we define properties a hierarchical tree distance measure should comply to, and describe our concrete implementation of these properties. Section 5 details how to embed the distance measure in a hierarchical multiclass classifier. The experiments of Section 6 compares this classifier with other classifiers. Finally, Section 7 presents our ideas for further research. Section 8 concludes.

2 Related Work

Dumais and Chen (Dumais and Chen, 2000) explore hierarchical classification of web content by ordering SVMs in a hierarchical fashion, and classifying based on user-specified thresholds. The authors focus on a two-level label hierarchy, as opposed to the 4-level UNSPSC hierarchy we utilize on in this paper. Assigning an instance to a class requires using the posterior probabilities propagated from the SVMs through the hierarchy. The authors conclude that exploiting the hierarchical structure of an underlying problem can, in some cases, produce a better classifier, especially in situations with a large number of labels.

Labrou and Finin (Labrou and Finin, 1999) use a global classifier based system to classify web pages into a 2-level DAG-based hierarchy of Yahoo! categories by computing the similarity between documents. The authors conclude that their system is not accurate enough to be suitable for automatic classification, and should be used in conjunction with active learning. This deviates from the method introduced in this paper in that model we introduce does not support DAGs and can be used without the aid of active learning, with promising results.

Wang, Zhou, and Liew (Wang et al., 1999) identify issues in local-approach hierarchical classification and propose a global-classifier based approach, aiming for closeness of hierarchy labels. The authors realize that the concern of simply being correct or wrong in hierarchical classification is not enough, and that only focusing on the broader, higher levels is where the structure, and thus accuracy, diminishes. To mitigate these issues, the authors implement a multilabel classifier based upon *rules* from features to classes found during training. These rules minimize a distance measure between two classes, and are deterministically found. Their distance measure is application-dependent, and the authors use the shortest distance between two labels. In this paper, we also construct a global classifier which aims to minimize distances between hierarchy labels.

(Weinberger and Chapelle, 2009) introduces a label embedding with respect to the hierarchical structure of the label tree. They build a global multiclass classifier based on the embedding. We utilize their method of classification with our novel distance measure.

Andersen et al. (Andersen et al., 2016) train a global classifier on the same UNSPSC dataset we have been given access to. The authors train a model using error correcting tournaments with logistic loss using the Vowpal Wabbit learning system, ignoring any hierarchical structure (Agarwal et al., 2011; Beygelzimer et al., 2009). We partially replicate their model and use it as a baseline comparison for our contribution.

3 Hierarchical Classification

The hierarchical structure among labels allows us to reason about different degrees of misclassification.

We are concerned with predicting the label of datapoints within a hierarchical taxonomy. We define the input data as a set of n tuples, such that a dataset D is defined by

$$D = \{(\mathbf{x}, y) \mid \mathbf{x} \in \mathbf{X}, y \in Y\}, \qquad (1)$$

where $\mathbf{x}$ is a q-dimensional datapoint in feature space $\mathbf{X}$ and y is a label in a hierarchically structured set of labels $Y = \{1, 2, \ldots, m\}$.

Assume we have a datapoint $\mathbf{x}$ with label $y = U$ from the label tree in Fig. 1. It makes sense that a

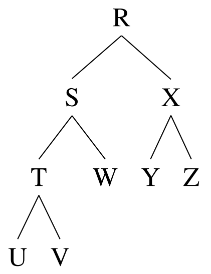


Figure 1: An example of a 3-level label tree.

prediction $\hat{y} = V$ should be penalized less than a prediction $\hat{y}' = Z$, since it is closer to the true label y in the label tree. We capture this notion of distance between any two labels with our hierarchy embracing distance measure, properties of which are defined in Section 4.

One commonly used distance measure is to count the number of edges on a path between two labels in the node hierarchy. We call this method the *Edges Between (EB)* distance. In Section 4.3, we construct a new distance measure based upon properties introduced in the following section. We denote this distance measure as the *AKM* distance.

4 Hierarchical Distance Measure

In the following, the hierarchical distance measure used for hierarchical classification is introduced. Section 4.1 introduces the notation used in this section. In Section 4.2 we reason about the properties of hierarchical distance measures. Finally, in Section 4.3, the AKM distance measure is formalized.

4.1 Notation

In interest of concise property definitions, we introduce the following notation for hierarchical label trees:

- R is the root node of a label tree.
- $\rho^i(A)$ is the i'th ancestor of A, such that $\rho^0(A)$ is A itself, $\rho^1(A)$ is the parent of A, and $\rho^2(A)$ is the grandparent of A, etc. We use $\rho(A)$ as shorthand for $\rho^1(A)$.
- $\text{ch}(A)$ is the set of children of A.
- $\text{sib}(A)$ is the set of siblings of A.
- $h(A)$ is the tree level of node A, where $h(R) = 0$ and $h(A) = h(\rho(A)) + 1$ when $A \neq R$.
- $\sigma(A, B)$ is the set of nodes on the path between nodes A and B, including both A and B. If $A = B$, $\sigma(A, B) = \{A\} = \{B\}$.
- $\alpha(A) = \sigma(A, R)$ defines the ancestors of node A.
- $\pi(A, B)$ is the set of edges on the path between A and B. Notice that for any edge, we always write the parent node first. For example, for the tree in Fig. 1, we have $\pi(U, W) = \{(T, U), (S, T), (S, W)\}$.
- We define $\text{sign}(x)$ for $x \in \mathbb{R}$ to return either -1, 0, or 1, depending on whether x is smaller than, equal to, or larger than 0, respectively.

Finally, we define a notion of structural equivalence between two nodes in a label tree, denoted $A \equiv B$, such that the root is structurally equivalent to itself, and

$$A \equiv B \iff (|\text{sib}(A)| = |\text{sib}(B)| \wedge \rho(A) \equiv \rho(B)) .$$

This recursive definition causes two nodes A and B to be structurally equivalent if all nodes met on the path from A to the root, pair-wise have the same number of children as the path from B to the root. For example, in Fig. 1 we have that that $T \equiv Z$. Notice in Fig. 2 how $B \not\equiv F$, due to the different number of siblings.

4.2 Properties

In the following, we reason about properties we think a tree distance measure should possess. We break properties a distance measure should adhere to, into two types: metric, and hierarchical.

4.2.1 Metric Properties

A distance function d is a metric if it satisfies the following four properties.

Property 1 (non-negativity). $d(A, B) \geq 0$

Property 2 (identity of indiscernibles).

$$d(A, B) = 0 \iff A = B$$

Property 3 (symmetry). $d(A, B) = d(B, A)$

Property 4 (triangle inequality).

$$d(A, C) \leq d(A, B) + d(B, C)$$

It can be shown that both the *EB* and the *AKM* distances satisfy these properties, and are thus metrics.

4.2.2 Hierarchical Properties

Besides the standard metric properties above, we propose three additional properties a distance measure should satisfy for the UNSPSC hierarchy.

Property 5 (subpath). *If a path can be split into two subpaths, its length is equal to the sum of the two subpaths' lengths. Formally, this property is stated as*

$$B \in \sigma(A, C) \implies d(A, C) = d(A, B) + d(B, C).$$

To exemplify the subpath property, in Fig. 1, we have that $d(U, W) = d(U, S) + d(S, W)$. Notice that this property is different from Property 4 (triangle inequality), since it requires the two subpaths $\pi(A, B)$ and $\pi(B, C)$ to be non-overlapping. It also implies a stronger result, namely that the distances $d(A, C)$ and $d(A, B) + d(B, C)$ are strictly equal.

Property 6 (child relatedness). *Consider a node in a label hierarchy with k children, and a datapoint x for which we wish to predict a label that is known to be among one of the k children. Intuitively, it should be easier to predict the correct label if there are fewer children (labels) to choose from. In other words, we say that the distance between two siblings should decrease with an increasing number of siblings. Formally, we capture this intuition with the following property*

$$A \equiv X \wedge A = \rho(B) \wedge X = \rho(Y) \implies$$
$$\text{sign}(|\text{ch}(A)| - |\text{ch}(X)|) = \text{sign}(d(X, Y) - d(A, B)).$$

Notice that if we let $X = A$, we get that a node is equally distant from any of its children. By the subpath property, this also implies that a node is equally distant from all of its siblings. For two structurally equivalent nodes A and X, the node with most children will have the shortest distance to any of its children. If they have the same number of children, the distance between any of the two nodes and a child is the same. For example, in Fig. 1 we have that S and X are structurally equivalent, having the same number of children. Thus, the property implies that $d(S, T) = d(X, Y)$.

Property 7 (common ancestor). *A prediction error that occurs at higher level in the tree should be more significant than an error occurring at a lower level. This is because that once an error occurs at some level, every level below will also contain errors. The levels above it may however still be a match. Therefore, it is desirable to have the first error occur as far as possible down the tree. Formally defined as*

$$X \in \alpha(A) \cap \alpha(B) \wedge X \notin \alpha(C) \implies d(A, B) < d(A, C).$$

In other words, if the nodes A and B match at some level indicated by X, for which A and C do not match, then A and B are more similar than A and C. For example, in Fig. 1 we have that U and W are more similar than U and X because U and W share an ancestor further down in the hierarchy than U and X.

4.3 AKM Distance

In the following we propose a new distance measure, the AKM distance, that satisfies the seven properties mentioned in Section 4.2.

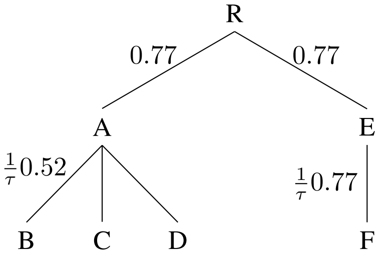


Figure 2: Example of edge weights used by d_{AKM}.

We define the AKM distance as a distance measure between nodes in a label tree:

$$d_{\text{AKM}}(A, B) = \sum_{(X,Y) \in \pi(A,B)} w(X, Y)$$

where

$$w(X, Y) = \begin{cases} \frac{1}{\log|\text{ch}(X)|+1} & \text{if } X \text{ is root} \\ \frac{1}{\tau} \cdot \frac{w(\rho(X),\, X)}{\log|\text{ch}(X)|+1} & \text{otherwise.} \end{cases}$$

This implies that dissimilarities at lower levels in the tree are deemed less significant for values of τ greater than 1. Also, due to the term $\log|\text{ch}(X)| + 1$, the distance between two siblings decreases logarithmically as more siblings are added. This prevents nodes with many children having a deciding impact on the weight between two nodes. We use base 10 logarithm, which affects Property 7 to be satisfied only when $\tau > 2.54$. Figure 2 shows an example label tree with edge weights as defined by AKM. The distance between nodes B and F is

$$\begin{aligned} d_{\text{AKM}}(B, F) &= w(R, A) + w(A, B) + w(R, E) + w(E, F) \\ &= \frac{1}{\log 2 + 1} + \frac{1}{\tau} \frac{\frac{1}{\log 2+1}}{\log 3 + 1} + \frac{1}{\log 2 + 1} \\ &+ \frac{1}{\tau} \frac{\frac{1}{\log 2+1}}{\log 1 + 1} \end{aligned}$$

5 Embedding Classification

As mentioned in Section 2, Weinberger et al. propose a method for classification that aims to minimize an arbitrary hierarchical distance measure between predicted and actual labels. In this section, we show how this embedding is created, and how it can be used for classification. For details, we refer to (Weinberger and Chapelle, 2009).

The hierarchical distance between labels is captured in a distance matrix $\mathbf{C}$. $\mathbf{C}$ is embedded such that the Euclidean distance between two embedded labels is close to their hierarchical distance. This embedding, $\mathbf{P}$, is defined as follows

$$\mathbf{P}_{mds} = \arg\min_{\mathbf{P}} \sum_{\alpha,\beta \in \mathbf{Y}} \left(\|\mathbf{p}_\alpha - \mathbf{p}_\beta\| - \mathbf{C}_{\alpha,\beta} \right)^2, \tag{2}$$

where the matrix $\mathbf{P} = [\mathbf{p}_\alpha, \ldots, \mathbf{p_c}] \in \mathcal{R}^{k \times c}$, the vector $\mathbf{p}_\alpha$ represents the embedding of label $\alpha \in \mathbf{Y}$, and $k \leq c$ is the number of dimensions.

From the embedding, a multi-output regressor can be learned which maps datapoints $\mathbf{x} \in \mathbf{X}$ with label $y \in Y$ to an embedded label using

$$\mathbf{W} = \arg\min_{\mathbf{W}} \sum_{(\mathbf{x},y) \in D} \|\mathbf{Wx} - \mathbf{p_y}\| + \lambda \|\mathbf{W}\|, \tag{3}$$

given the embedding and the regressor, future datapoints $\hat{\mathbf{x}}$ can be classified in the following way

$$\hat{y} = \arg\min_{\alpha \in Y} \|\mathbf{p}_\alpha - \mathbf{W}\hat{\mathbf{x}}\|. \tag{4}$$

Note that the way we classify differs slightly from the method of Weinberger et al., as we reduce the dimensions of $\mathbf{P}_{mds}$ to k. In the following, we will refer to the above type of classifier, which can embed a metric distance measure, as an embedding classifier (EC).

6 Experiments

In the following, we compare three different types of classifiers. The first is a standard multiclass logistic regression classifier that, given a datapoint, predicts a label without accounting for any hierarchy amongst labels. Two other classifiers are built, both of which are based on distance matrices described Section 5. We call these classifiers the *EB-EC*, and *AKM-EC*, where their corresponding $\mathbf{C}$ matrices represent the EB and AKM distances, respectively.

6.1 Dataset

We have access to a dataset of 805,574 invoice lines, each representing the purchase of an item. Each invoice line is represented by 10 properties, such as issue date, due date, item name, description, price, quantity, seller name, etc. In total, 721,663 of the invoice lines have assigned a useful UNSPSC version 7.0401 label. UNSPSC is a 4-level hierarchical taxonomy, consisting of the levels *Segment*, *Class*, *Family*, and *Commodity*, mentioned in order of increasing specificity. For simplicity, we will refer to these levels as *level 1*, *level 2*, *level 3*, and *level 4*, respectively. Not all datapoints contain a label at the lowest hierarchy level (*level 4*). The distribution of labels among the 4 levels is roughly as follows: at least on level 1: 100 %, at least on level 2: 96 %, at least on level 3: 83 %, at least on level 4: 60 %. We split the dataset into a 70/30 training and test set, and then randomly shuffle each. Datapoints from the test set have been removed if their label is not present in the training set, resulting in a train and test set split of 522,000 and 199,663 datapoints, respectively, which we use for all further experiments.

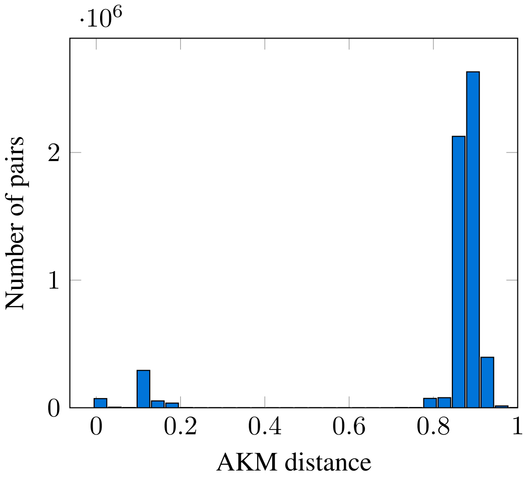


Figure 3: Histogram over the distribution of AKM-distances between any two UNSPSC labels for $\tau = 3$.

Even though UNSPSC version 7.0401 contains 20,739 unique labels (including non-leaves), the dataset includes only 3400 unique labels.

Figure 3 shows a histogram over $d_{\text{AKM}}(y_i, y_j)$ distances for all pairs of labels in the dataset. There are a total of 50 buckets each distance can fall into. Increasing τ simply narrows the interval of the distances, which is expected, due to τ appearing in the denominator in the definition of the AKM distance. This figure shows that the distances between labels are grouped into two groups. A path between two labels that passes through the root incurs an AKM distance of at least 0.73, independently of τ as level 1 contains 55 labels. This means that label pairs in the smaller, leftmost group share an ancestor different from the root. Those in the rightmost, larger group, have a path that crosses through the root.

6.2 Results on UNSPSC Dataset

In this section, we formulate and discuss results from experiments, on the UNSPSC-labeled invoice lines dataset.

6.2.1 Choosing Features

We use a univariate feature selection method as described in (Chen and Lin, 2006) to test the discrim-

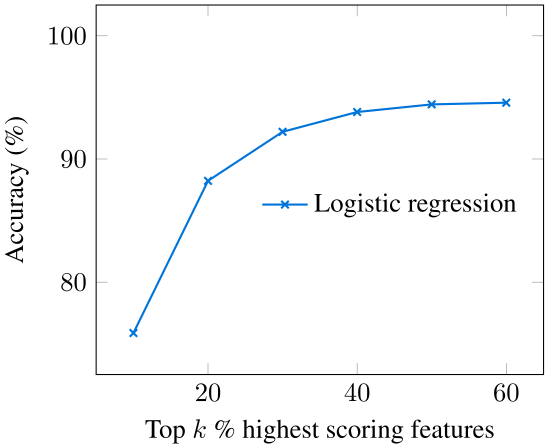


Figure 4: Accuracy across feature percentages.

inative power of different subsets of features. The method calculates an F-score for each feature and, by choosing the top k scoring features (those with highest F-values), a feature set is constructed from which a predictive model is built. Figure 4 shows the accuracy achieved on different feature sets using a standard multiclass logistic regression classifier on the test set. All logistic regression classifiers used in this paper have been made with the scikit-learn package (Pedregosa et al., 2011).

It is evident that greater accuracies are achieved with more features. However, it does not seem like the accuracy will improve much beyond the top $40\,\%$ features. Therefore, as a trade-off between high accuracy and time spent running tests, we use the top $40\,\%$ of features for further experiments.

6.2.2 τ Test

The τ value used in the definition of the AKM distance impacts to which extent higher level errors are considered more costly than lower level errors. We define the term *level k matches* to be the number of datapoints that are correctly predicted at the k'th level, out of the total number of datapoints that have a label at the k'th level or below in the hierarchy. For example, if a classifier predicts the label correct at the 2nd level for 4500 out of 10,000 samples, the amount of level 2 matches is $4500/(10{,}000 \cdot 0.96) = 43.2\,\%$, since only $96\,\%$ of the dataset has a label at the 2nd or below. Figure 5 uses this measure to show that a larger value of τ does in fact decrease the prediction accuracy at lower levels, but slightly increases accuracy at the highest level. This is expected, as a higher values of τ lowers the importance of accuracy at lower levels.

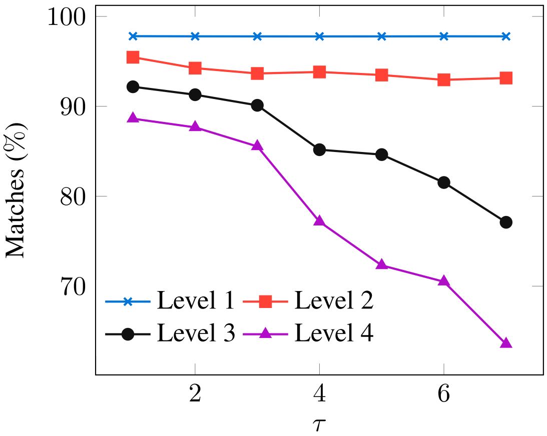


Figure 5: Level matches of AKM-EC in relation to τ.

6.2.3 Proof of Concept

Embedding classifiers are constructed to minimize their respective distances between predicted and actual labels in the dataset. When predicting the labels of a test dataset, our hypothesis is that they achieve a lower respective distance between the predicted and actual labels, compared to that of logistic regression. Figure 6 compares the logistic regression classifier to AKM-EC and EB-EC, according to their respective distance measures. The figure shows that they are very similar, and this is due to the fact that logistic regression have high accuracy, as shown in Fig. 4. The high accuracy results in a low average distance, as a correct prediction yields a distance of 0.

It is therefore interesting to isolate misclassifications to compare how far the four classifiers deviate on errors. Figure 7 plots the average AKM and EB distances for each classifier across dimensions on misclassified datapoints only. In this figure, we see that AKM-EC minimizes the average AKM distance, and the EB-EC minimizes the average EB distance, as expected. Notice how the EB and AKM distances follow a similar pattern. This is also expected, as the distances are very similar. AKM-EC and EB-EC have lower distances than logistic regression, because they optimize for their hierarchical distance measures. Since logistic regression receive a loss of 1 on misclassifications, we see that it perform worse on each distance measure.

We note that for misclassifications, the average AKM distance between the predicted and correct class for the AKM-EC classifier is 0.213, while the average AKM distance for logistic regression is 0.348. In order to better understand the meaning of these numbers, we calculate the percent of non-root crossing miss classifications. For AKM-EC, this is $89\,\%$, meaning only $11\,\%$ of misclassifications have a path from the predicted class to the actual class that crosses the root

Metric	AKM-EC	Log. reg.
Avg. AKM-distance	0.219	0.24
AKM-distance on miss	0.72	0.88
% correct classifications	70 %	73 %
% root-crossing on errors	47 %	63 %

Table 1: Classifier comparison on the Twenty Newsgroups dataset between the AKM-EC and logistic regression classifiers.

node in the hierarchy. For logistic regression this is 72 %, meaning 28 % of misclassifications have a path through the root node.

6.3 Verification

In order to verify the properties of the AKM distance measure, we compare the AKM-EC classifier to the logistic regression classifier on the Twenty Newsgroups dataset (Lichman, 2013), containing 11,314 datapoints for training and 7532 datapoints for testing. The results show that training a classifier with respect to the AKM distance ensures a lower AKM distance, as shown in Table 1, but lower exact correctness. This is not necessarily a problem, due to the AKM-EC classifier's superior ability to make accurate guesses.

7 Future Work

In this section, we consider topics that could be addressed in future work.

Implementing a weighted algorithm when embedding the $\mathbf{C}$ matrix could possibly improve the accuracy of the embedding classifiers. This is due to the fact that, currently, label embeddings are independent of how many datapoints of that label exist, possibly resulting in labels with few samples causing noise.

Since the embedding classifier uses many linear regressors, weighting the importance of each regressor's output in relation to its loss could possibly benefit classification.

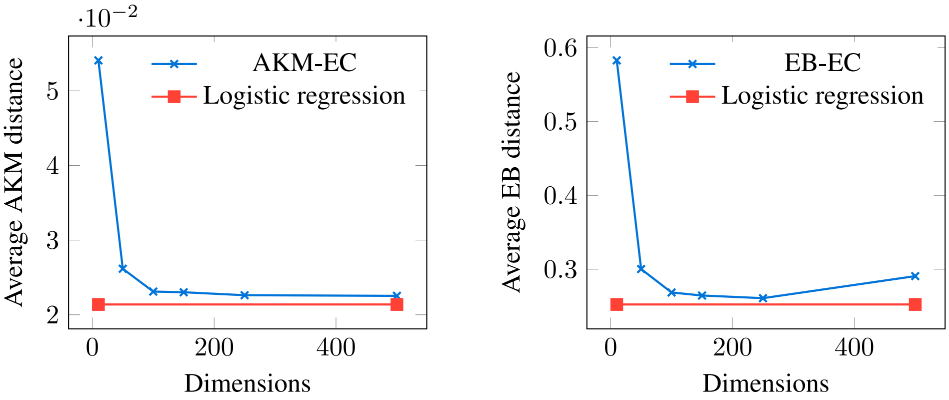
There are also different ways of formulating a distance measure such that they are vastly different than the AKM and EB measures, which take into account properties of other hierarchies. It would be interesting to evaluate how well the embedding classifier manages to embed these distance measures.

Expanding the model to support directed acyclic graphs (DAGs) could open the model to supporting other types of hierarchical datasets.

One could consider expanding the 4-level UNSPSC tree to five or more levels. The hierarchy used in our experiments has 55 children below the root level, making the choice of path from the root difficult.

8 Conclusion

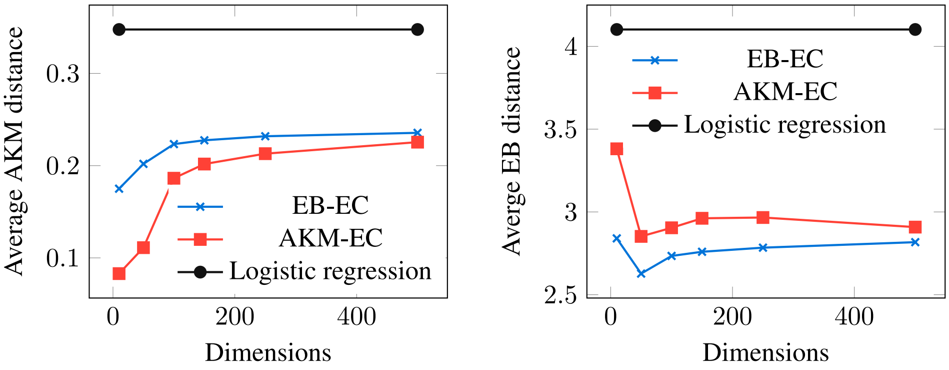
We have introduced a novel hierarchical distance measure that outperforms logistic loss on misclassifications. This measure fulfills the intuitive properties for the UNSPSC product and service taxonomy. To take advantage of this distance measure, we propose an embedding classifier, that embeds matrices representing hierarchical distances to a lower-dimensional space. In this space, datapoints are mapped to an embedded class, and predictions are made. This classifier can be combined with other distance measures. Experiments suggest that the distance measure may not be the best for classification, but performance on misclassified samples is promising. Accuracy is traded for lower hierarchical distance on misclassifications. This is useful in the UNSPSC hierarchy, where erroneous predictions can vastly change the context of a label.



(a) Average AKM distance comparison for $\tau = 3$.

(b) Average EB distance comparison.

Figure 6: Average AKM and EB distances for four different classifiers on the test dataset.



(a) Average AKM distances for $\tau = 3$.

(b) Average EB distances across classifiers.

Figure 7: Average AKM and EB distances of embedding classifiers on the test dataset for misclassified samples.

REFERENCES

Agarwal, A., Chapelle, O., Dudik, M., and Langford, J. (2011). A reliable effective terascale linear learning system. *Computing Research Repository*, abs/1110.4198.

Andersen, M. M., Caspersen, K. M., Eriksen, A. B., Madsen, M. A., and Madsen, M. B. (2016). Automatic categorization of invoice lines.

Beygelzimer, A., Langford, J., and D. Ravikumar, P. (2009). Error-correcting tournaments. *Computing Research Repository*, abs/0902.3176.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.

Chen, Y.-W. and Lin, C.-J. (2006). *Feature Extraction: Foundations and Applications*, chapter Combining SVMs with Various Feature Selection Strategies, pages 315 – 324. Springer Berlin Heidelberg, Berlin, Heidelberg.

Dumais, S. and Chen, H. (2000). Hierarchical classification of web content. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 256 – 263. ACM.

Labrou, Y. and Finin, T. (1999). Yahoo! as an ontology: using yahoo! categories to describe documents. In *Proceedings of the eighth international conference on Information and knowledge management*, pages 180 – 187. ACM.

Lichman, M. (2013). UCI machine learning repository.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Programme, U. N. D. (2016). United nations standard products and services code homepage.

Silla, Carlos N., J. and Freitas, A. A. (2011). A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22(1 – 2):31 – 72.

Wang, K., Zhou, S., and Liew, S. C. (1999). Building hierarchical classifiers using class proximity. In *Proceedings of the 25th International Conference on Very Large Data Bases*, pages 363–374, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Weinberger, K. Q. and Chapelle, O. (2009). Large margin taxonomy embedding for document categorization. In Koller, D., Schuurmans, D., Bengio, Y., and Bottou, L., editors, *Advances in Neural Information Processing Systems 21*, pages 1737–1744. Curran Associates, Inc.