Noname manuscript No.
(will be inserted by the editor)

Counts-of-Counts Similarity for Prediction and Search in Relational Data

Manfred Jaeger · Marco Lippi · Giovanni Pellegrini · Andrea Passerini

the date of receipt and acceptance should be inserted later

Abstract Defining appropriate distance functions is a crucial aspect of effective and efficient similarity-based prediction and retrieval. Relational data are especially challenging in this regard. By viewing relational data as multi-relational graphs, one can easily see that a distance between a pair of nodes can be defined in terms of a virtually unlimited class of features, including node attributes, attributes of node neighbors, structural aspects of the node neighborhood and arbitrary combinations of these properties. In this paper we propose a rich and flexible class of metrics on graph entities based on earth mover's distance applied to a hierarchy of complex counts-of-counts statistics. We further propose an approximate version of the distance using sums of marginal earth mover's distances. We show that the approximation is correct for many cases of practical interest and allows efficient nearest-neighbor retrieval when combined with a simple metric tree data structure. An experimental evaluation on two real-world scenarios highlights the flexibility of our framework for designing metrics representing different notions of similarity. Substantial improvements in similarity-based prediction are reported when compared to solutions based on state-of-the-art graph kernels.

1 Introduction

Nearest-neighbor search is a fundamental problem that appears in many different contexts. In machine learning, the labels of nearest neighbors for a test case are used to predict its unknown label. In information retrieval, nearest neighbors of a query object are returned as the most relevant matches for the query. In all contexts, there are two

M. Jaeger · A. Passerini
Institut for Datalogi, Aalborg University

M. Lippi
Dipartimento di Scienze e Metodi per l'Ingegneria, Università degli Studi di Modena e Reggio Emilia

G. Pellegrini · A. Passerini
Dipartimento di Ingegneria e Scienza dell'Informazione, Università degli Studi di Trento.

E-mail: jaeger@cs.aau.dk, marco.lippi@unimore.it, giovanni.pellegrini@unitn.it, andrea.passerini@unitn.it

important aspects to nearest neighbor approaches: first, a suitable distance measure has to be defined, so that nearby neighbors according to this metric provide accurate classifications, or relevant retrieval results. Second, for the given metric, suitable data structures and algorithms have to be designed for an efficient retrieval of the nearest neighbors of a query object.

In this paper we consider similarity and search in attributed, multi-relational graphs, which provide a useful, common abstraction level for relational databases on the one hand, and network data on the other. For a given query object (node), we want to find similar objects in the data graph. Our first major concern is to define a powerful and flexible framework for defining distance metrics. The particular challenge of graph data lies in the fact that two nodes may be compared based on an essentially unlimited and complex class of features: one can take into consideration the attributes of the nodes itself, attributes of graph neighbors, structural properties of the graph neighborhood, where for the latter one can take into account neighborhoods of different radii around the nodes being compared.

Our approach is distinguished by the fact that it takes complex quantitative aspects of relational neighborhoods into account. Specifically, we base our distance metrics on *counts-of-counts* features we introduced in (Jaeger et al, 2013), that represent detailed quantitative information about relational neighborhoods. To illustrate the basic idea of counts-of-counts features, consider the case of a bibliographic database containing *authors* and *papers*, an *author_of* relation between authors and papers, and a *cites* relation between papers. We may then compare different persons in the database based on their citation profile, perhaps using a bibliometric index like h-index or i10-index. In both cases, the comparison is based on the counts-of-counts statistic that for each possible count of citations i provides the count $n(i)$ of papers with i citations. Note that in this paper we use the term "counts-of-counts" for an arbitrary nested structure of counts, from simple counts (e.g. the number of papers written by an author), to multiple nesting levels of counts (an example for three levels of counts will be given below in Section 8.1.1 and Figure 8(a)).

Measuring similarity of authors based on a bibliometric profile will be appropriate in certain contexts, e.g., when evaluating candidates for an academic position. In other contexts, such as searching for potential scientific collaborators, a similarity measure based on the number of publications in certain subject areas will be more useful. In all cases, however, we focus on scenarios where the absolute numbers of related entities matter. This contrasts with scenarios where only the existence of certain relationships matter, which can be captured by logical rules with existential quantification such as: classify a person as an academic, if there exists a scientific article of which the person is an author.

In this paper we introduce a very flexible framework for specifying customized notions of similarity in graph data. The framework first supports to specify what counts-of-counts statistics should be considered, and then provides a parametric class of distance metrics on the given statistics. Figure 1 illustrates an application of our approach. Here we are considering a movie database containing information on movies and actors. We first are interested in actor similarity based on the genre profile of the movies they have appeared in. The first row in the figure contains graphical representations of the relevant relational neighborhood data for this purpose: the central node is the actor in question, the inner circle of blue nodes represents the movies the actor had a role in, and the outer colored nodes represent the different genres with which the movies are labeled. In total, 20 different genres are considered, each represented by a different color.

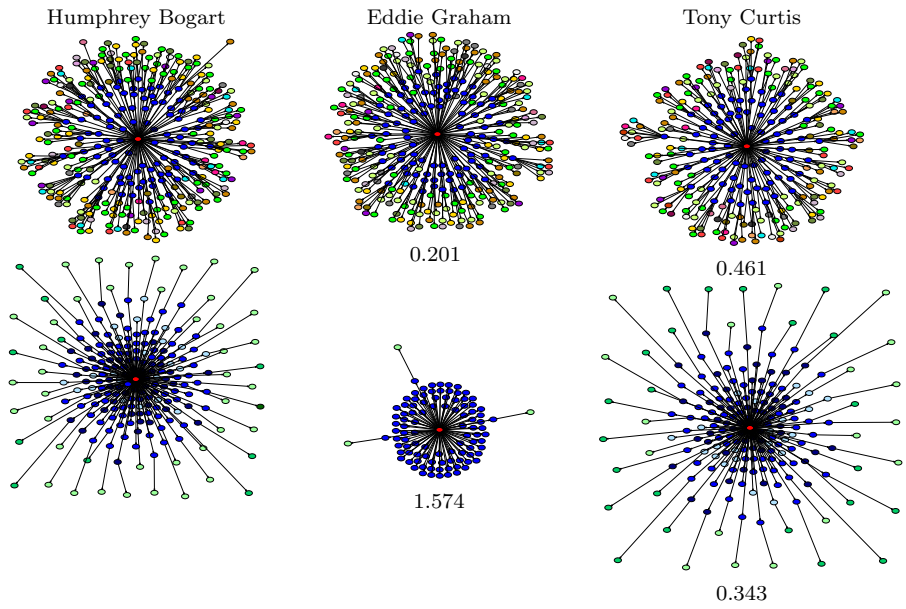


Fig. 1 Nearest neighbors of Humphrey Bogart according to genre-/business-based similarity.

A movie can be associated with more than one genre, and our metric will distinguish, e.g., between the case where an actor has appeared in a movie that is both a *romance* and a *comedy*, and the case where the actor has appeared in a movie that is a *romance*, and another one that is a *comedy*. Based on this similarity concept, the most similar actor found for the query actor Humphrey Bogart is Eddie Graham. This relatively little known actor has a very similar profile to Bogart in terms of the number of movies and their genres. We next consider similarity based on financial/business characteristics of an actor. The bottom row of Figure 1 illustrates relevant relational neighborhood data selected for this purpose: central nodes are again the actor; the inner blue nodes are again the movies the actor has appeared in, now shaded dark/light/medium according to whether the actor had a leading/supporting/unspecified role in the movie. Movies are connected to a *budget* attribute represented by a dark/medium/light green node according to whether the movie had a large/medium/small budget (budget data is not available for all movies, so movies need not be connected to a budget node). In terms of these business statistics, Graham now is rather dissimilar to Bogart, since the status of his roles are all unspecified, and there is no budget data for most of his movies. In terms of business similarity, the closest actor for Bogart now is Tony Curtis. The numbers underneath the graphs for Graham and Curtis are the actual distances to Bogart according to our genre-based and business-based metrics.

We use our earlier *type extension tree* framework (Jaeger et al, 2013) for specifying suitable counts-of-counts features. This paper makes the following new contributions:

– We introduce a parametric class of *logistic evaluation functions* that transfoms a combinatorial counts-of-counts value into a hierarchical numerical structure. This transformation can be understood as the evaluation of a neural network, and we exploit this analogy by using backpropagation techniques to learn the parameters of the evaluation function.

- We show how to approximate the hierarchical numerical structure obtained from the logistic evaluation function by a collection of multi-dimensional histograms, and define an earth-mover's-distance based metric on these histograms. Via a suitable choice of the underlying counts-of-counts features, and adaptation of the parameterization of the evaluation function, this metric becomes highly flexible and adaptable for different datasets and applications.
- We introduce an approximation to the earth-mover's distance on multi-dimensional histograms using sums of marginal distances, and show that for many instances of our metrics the approximation is exact.
- It is shown how the approximate distance computation in conjunction with a metric tree data structure supports efficient retrieval of nearest neighbors.
- An experimental evaluation shows that the introduced framework is efficient and effective for nearest neighbor retrieval under diverse notions of similarity, and allows to address similarity-based prediction tasks with substantial improvements over alternative similarity-based approaches.

A simple precursor of the metric introduced in this paper was already described in (Jaeger et al, 2013). That metric was defined directly on the raw counts-of-counts feature values, and did not permit the kind of problem-specific adaptation we now obtain through the logistic evaluation function.

2 Counts-of-counts features

Our approach to defining similarity between graph objects is feature based: we use a highly expressive and flexible framework for defining features for graph entities, and then compare entities by comparing their feature values, based on suitable feature metrics. For feature definition, we use the *type extension tree (TET)* framework, which we introduced in (Jaeger et al, 2013).

Type extension trees represent complex "counts-of-counts" features that describe the combinatorial structure of a graph entity's neighborhood. A TET feature defines which relations to follow for assembling the relevant neighborhood, and what attributes of neighboring nodes to consider. In the following we summarize in a somewhat streamlined and simplified form the precise definitions given in (Jaeger et al, 2013).

The data model underlying our approach is that of an *attributed, multi-relational* graph, which is given by a set of entities (nodes) E, a set of attributes $A = \{a_1, \ldots, a_l\}$ defined on the entities, and binary relations $R = \{r_1, \ldots, r_m\}$ between the entities. For the purpose of this paper we make the restriction that all attributes and relations are Boolean. Generalizations to arbitrary categorical and especially numerical attributes and relations are not difficult in principle, but are omitted for now.

If e, e' are entities, then $a_i(e)$ and $r_j(e, e')$ are *ground atomic statements* or simply *ground atoms* that are either *true* or *false* for a given data graph. An *atomic statement* or *atom* is an expression of the form $a_i(X)$ or $r_j(X, Y)$, where X, Y are variable symbols. We remark that the graphs shown in Figure 1 are not data graphs or subgraphs of a data graph, because the nodes in these graphs do not represent entities, but ground atoms. Thus, for example, one of the colored nodes on the periphery of Bogart's genre graph stands for the ground atom `drama`(*Casablanca*).

Definition 1 A *type extension tree* (TET) is a rooted tree whose nodes are labeled with atoms, and whose edges can be labeled with variables.

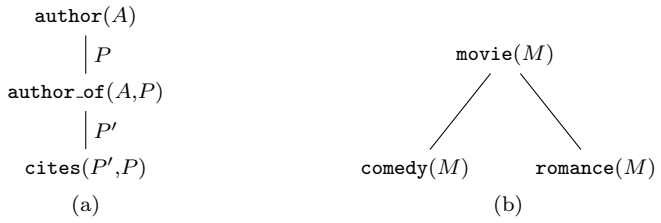


Fig. 2 TET examples

To simplify notation and subsequent definitions, we here give a slightly simplified definition of TETs. The more general definition of (Jaeger et al, 2013) also allows for conjunctions of atoms at the nodes of the tree, and we will make use of this option in one of our experiments in Section 8.3.3.

Figure 2 shows two examples of TETs. TET (a) is for bibliographic data that includes the attribute `author`, and the relations `author_of` and `cites`. This TET is also shown in Figure 3 (a), which in (b) shows the data sub-graph of the relational neighborhood of an author (blue (dark) central node) with his/her authored papers (5 inner yellow (light) nodes connected by solid edges to author), and other papers citing these (outer yellow (light) nodes connected by dashed edges). The value of the TET feature (to be formally defined below) will provide a complete picture of this sub-graph's structure.

The TET in Figure 2 (b) is for a movie database that contains the attribute `movie` and genre attributes `comedy` and `romance`. This movie TET does not contain any relations, or variables as labels on the edges. It defines a feature for movie entities, only as a function of the entities' attribute values. The feature value corresponds to a complete cross-classification of whether the movie is a comedy and/or a romance. The pure counts-of-counts features as represented by Figure 2 (a), and attribute features represented by Figure 2 (b) can be combined and nested in arbitrary ways. Examples of such combinations will be used in our experiments (cf. Figure 8 (b) and (c)).

Type extension trees are evaluated for graph entities. A TET may define a feature for single entities, for pairs of entities, or any number of entities. The "arity" of a TET is determined by the number of free variables contained in the TET, where a *free variable* is any variable that appears in an atom at one of the nodes v, such that this variable does not appear as the label on any edge on the path from v to the root. In particular, all the variables appearing at the root of a TET are free, and very often these are precisely the free variables of the TET. In Figure 2, (a) has a single free variable A, and (b) the single free variable M. Any sub-tree of a TET is itself a TET. The sub-tree

$$\texttt{author\_of}(A, P) \xrightarrow{P'} \texttt{cites}(P', P) \tag{1}$$

has two free variables A, P. A TET with k free variables defines a feature for k-tuples of entities. Thus, both TETs in Figure 2 define features for single entities (of type author, respectively movie). Sub-tree (1) defines a feature for pairs of author and paper entities.

In the following definition we consider tuples of variables $\boldsymbol{X} = X_1, \ldots, X_k$, and corresponding tuples $\boldsymbol{e} = e_1, \ldots, e_k$ of graph entities. Given a subset $\boldsymbol{Z} \subseteq \boldsymbol{X}$ of variables, we then denote by $\boldsymbol{e}[\boldsymbol{Z}]$ the corresponding selection of elements from $\boldsymbol{e}$. We use $\alpha(\boldsymbol{X})$ as a generic expression for an atom of either the form $a_i(X)$ or $r_j(X, Y)$. The definition

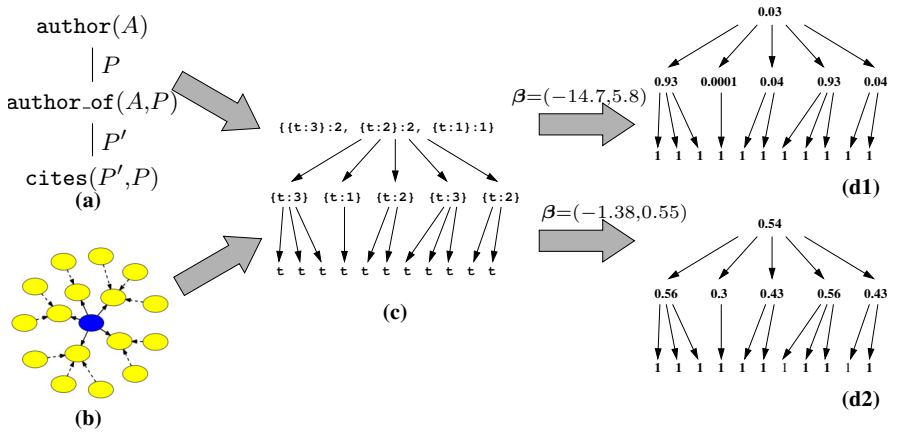


Fig. 3 Overview: illustration of Definitions 2, 3 and 4. (a): TET specification $T(A)$; (b): relational neighborhood of example author a (blue (dark) node); (c): TET value $V(T(a))$ (Definition 2); (d1),(d2): two different numeric evaluations of $V(T(a))$ (Definition 4) with different parameterizations $\boldsymbol{\beta}$ (Definition 3)

now defines the counts-of-counts feature value defined by a TET for a specific tuple $\boldsymbol{e}$ of entities. The precise mathematical structure that encodes our "counts-of-counts" values are nested multisets, where counts now correspond to the multiplicities with which a given element (itself potentially a nested multiset) occurs in a multiset. Thus, TET values according to the following definition are complex, structured objects, not simple scalars (distilling these complex values into scalars will be the task we address in Section 3) .

Definition 2 Let $T(\boldsymbol{X})$ be a TET with free variables $\boldsymbol{X} = X_1, \ldots, X_k$. Let $\boldsymbol{e} = e_1, \ldots, e_k$ be a k-tuple of graph entities. The value of the TET feature $T(\boldsymbol{X})$ for $\boldsymbol{e}$, denoted $V(T(\boldsymbol{e}))$, is inductively defined as follows:

i Let $\alpha(\boldsymbol{Z})$ ($\boldsymbol{Z} \subseteq \boldsymbol{X}$) be the atom at the root of T. If $\alpha(\boldsymbol{e}[\boldsymbol{Z}]) = \mathit{false}$, then $V(T(\boldsymbol{e})) = \mathit{false}$ (if the root of the tree evaluates to *false*, then there is no further recursive evaluation of the TET).

ii If $T(\boldsymbol{X})$ consists of the single node $\alpha(\boldsymbol{X})$, then $V(T(\boldsymbol{e})) = \alpha(\boldsymbol{e}) \in \{\mathit{true}, \mathit{false}\}$ (leaves are directly evaluated by the Boolean value of their atom).

iii If neither case i nor case ii applies, then $T(\boldsymbol{X})$ has $m \geq 1$ children that are roots of sub-trees $T_h(\boldsymbol{Z}_h)$ with free variables $\boldsymbol{Z}_h$ ($h = 1, \ldots, m$). $V(T(\boldsymbol{e}))$ then is defined as a tuple $(\mu_1, \ldots, \mu_m)$, where μ_h is a multiset of values of T_h. To define μ_h for sub-tree T_h we distinguish two cases:

- the edge leading to T_h is not labeled by a variable; then $\boldsymbol{Z}_h \subseteq \boldsymbol{X}$, and we define
$$\mu_h = \{V(T_h(\boldsymbol{e}[\boldsymbol{Z}_h])\}. \tag{2}$$
(the multiset contains only a single element in this case).
- the edge leading to T_h is labeled by a variable Y_h; then $\boldsymbol{Z}_h = (\tilde{\boldsymbol{Z}}_h, Y_h)$ for some $\tilde{\boldsymbol{Z}}_h \subseteq \boldsymbol{X}$, and we define
$$\mu_h = \{V(T_h(\boldsymbol{e}[\tilde{\boldsymbol{Z}}_h], e')) \mid e' \in E\} \tag{3}$$
This is to be understood as a multiset, counting the multiplicities of identical values obtained for different $e' \in E$.

We write $\{\gamma_1 : n_1, \ldots, \gamma_l : n_l\}$ to denote a multiset that contains n_i copies of the value γ_i $(i = 1, \ldots, l)$.

Example 1 Evaluating the TET $T(M)$ in Figure 2 (b) for an entity that is not a movie returns the value *false*. If m is a movie, then $V(T(m))$ is one of $(\{f\}\{f\})$, $(\{t\}\{f\})$, $(\{f\}\{t\})$, or $(\{t\}\{t\})$, depending on whether or not m is a comedy and/or a romance.

Example 2 To demonstrate a recursive value computation for the TET in Figure 2 (a) we first consider the sub-TET $T_1(A, P)$ shown in (1). This is evaluated for pairs of entities (a, p), and returns *false* if $\texttt{author\_of}(a, p)$ is false. Otherwise, $V(T_1(a, p))$ is $\{true : k, false : l\}$, where k is the number of papers p' citing p, and l is the number of entities p' for which $\texttt{cites}(p', p)$ is false. The number l includes the papers that do not cite p, and even domain entities that are not of type `paper`. The count of occurrences of *false* values is often semantically not very meaningful, and it will have no influence on the further processing of TET values that we introduce in the following sections. We therefore typically suppress *false* counts in the notation for TET values, and we here would abbreviate $\{true : k, false : l\}$ as $\{true : k\}$.

Turning to the evaluation of the full TET of Figure 2 (a) for an author a, we obtain the value $V(T(a))$ as the multiset $\{\{true : k_i\} : z_i\}$, where z_i is the number of papers p by author a with k_i citations (again omitting the count of *false* values generated by domain entities p for which $\texttt{author\_of}(a, p)$ is false).

Figure 3 (c) shows a tree representation of the value of the TET from Figure 2 (a) evaluated for the author a represented by the data sub-graph of Figure 3 (b). The tree shows the final value $V(T(a))$ at the root, and its decomposition into values of the sub-tree (1) (middle layer of the tree), and values of the leaf node $\texttt{cites}(P', P)$ (leaves of the tree). In this figure we also have omitted the *false* values. Including them would add to every node $\{t : k\}$ of the middle layer $n - k$ additional children labeled with f, where n is the total number of entities in the domain (authors and papers).

3 Logistic Evaluation Function

The evaluation of a TET $T(\boldsymbol{X})$ as $V(T(\boldsymbol{e}))$ as defined in Section 2 leads to a complex nested multiset structure of Boolean values. In (Jaeger et al, 2013), we presented two approaches for using these values as the basis for prediction tasks and similarity measures:

- by defining a *discriminant function* f that maps TET values $V(T(\boldsymbol{e}))$ to real numbers, and that can be used for binary classification tasks;
- by defining a metric $d(V(T(\boldsymbol{e})), V(T(\boldsymbol{e}')))$ on TET values that can be used for distance-based methods such as nearest-neighbor prediction.

In (Jaeger et al, 2013), we tested both approaches on the artificial problem of classifying authors in a bibliographic database according to the binary attribute of whether their h-index is greater than 7. We note that this is an artificial task, since the h-index is a deterministic function of the data, and knowing the definition of h-index, one can always "predict" it with certainty. The challenge here is to "discover" the definition of the $(h > 7)$-attribute from examples of authors labeled as $(h > 7)$ or $(h \leq 7)$. In (Jaeger et al, 2013) we reported F1 scores of 61.3% and 91.2% for this

prediction task when using the discriminant function and nearest-neighbor prediction, respectively.

These results indicate that the discriminant function and metric definition of (Jaeger et al, 2013) are not flexible and powerful enough to fully exploit the information given by a TET value $V(T(a))$ to learn how to solve the $(h > 7)$ prediction task with 100% accuracy. We will greatly refine and unify these previous approaches by

- defining a rich class of evaluation functions $l^{\boldsymbol{\beta}}$ that map TET values to real numbers and that are parameterized by an adjustable vector $\boldsymbol{\beta}$;
- defining a metric on the nested multiset structure of real numbers that is generated by the recursive evaluation of $l^{\boldsymbol{\beta}}$ on a TET value $V(T(\boldsymbol{e}))$.

Thus, our evaluation functions $l^{\boldsymbol{\beta}}$ will be used directly as discriminant functions for prediction, and as the basis for defining metrics $d^{\boldsymbol{\beta}}$ that can be customized to represent specific similarity concepts by adjusting the parameters $\boldsymbol{\beta}$ of the underlying function.

We first define a parameterization of a TET:

Definition 3 Let $T(\boldsymbol{X})$ be a TET. A *weight assignment* $\boldsymbol{\beta}$ for T assigns a nonnegative real number to all non-leaf nodes and all edges of T. A weight assignment can be written as $(\beta_0^r, \beta_1^r, \ldots, \beta_m^r, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_m)$, where β_0^r is the weight assigned to the root, β_i^r is the weight assigned to the edge from the root to its ith child, and $\boldsymbol{\beta}_i$ is the weight assignment to the ith sub-tree.

Given a weight assignment for TET, we define a function on TET values γ via a recursive definition over the nested multiset structure of γ:

Definition 4 For a TET T with weight assignment $\boldsymbol{\beta}$ the *logistic evaluation* function $l^{\boldsymbol{\beta}}$ is defined as follows. Let $\gamma = V(T(\boldsymbol{e}))$ be a value.
Base cases:

- If $\gamma =$ *false*, define $l^{\boldsymbol{\beta}}(\gamma) := 0$.
- If $\gamma =$ *true*, define $l^{\boldsymbol{\beta}}(\gamma) := 1$.

Recursion:

- If $\gamma = (\mu_1, \ldots, \mu_m)$, with multisets μ_i of values of sub-trees T_i, define:

$$l^{\boldsymbol{\beta}}(\gamma) := \sigma\left(\beta_0^r + \sum_{i=1}^{m} \beta_i^r \sum_{\gamma' \in \mu_i} l^{\boldsymbol{\beta}_i}(\gamma')\right)$$

  where σ is the sigmoid function $\sigma(x) = 1/(1 + e^{-x})$.

The recursive logistic evaluation of a TET value γ leads to a nested multiset structure of real numbers that follows the structure of γ. We refer to this structure as the *logistic evaluation tree*, denoted $L^{\boldsymbol{\beta}}(\gamma)$.

Example 3 Consider the TET value γ shown in Figure 3 (c), and let $\boldsymbol{\beta}$ be the weight assignment that assigns $\beta_0 = -14.7$ to both non-leaf TET nodes, and $\beta_1 = 5.8$ to both edges. Thus, $\boldsymbol{\beta} = (-14.7, 5.8, (-14.7, 5.8))$ when properly expanded in the notation of Definition 3 (which is represented in a simplified manner in Figure 3). The evaluation of each leaf node representing a *true* value returns a 1. The leaf nodes labeled with *false* which are omitted in the figure all evalute to 0. Now consider a sub-value $\{t : 3\}$ at the middle level of γ. We obtain:

$$l^{(-14.7,5.8)}(\{t : 3\}) = \sigma(-14.7 + 5.8 \cdot (1 + 1 + 1)) = 0.937$$

The omitted *false* values have no impact on this calculation, since they would only add a number of 0 terms to the inner sum. Similarly, $l^{(-14.7,5.8)}(\{t:2\}) = 0.0431$, and $l^{(-14.7,5.8)}(\{t:1\}) = 0.0001$. Note that here, and in the following computation, we have $m = 1$, since the underlying TET has no branching. For the top-level evaluation we then obtain:

$$l^{(-14.7,5.8,(-14.7,5.8))}(\gamma) = \sigma(-14.7 + 5.8 \cdot (0.93 + 0.0001 + 0.04 + 0.93 + 0.04)) = 0.03$$

Figure 3 (d1) shows the logistic evaluation tree $L^{\boldsymbol{\beta}}(\gamma)$ induced by this computation of $l^{\boldsymbol{\beta}}(\gamma)$. The final value $l^{\boldsymbol{\beta}}(\gamma)$ is the root of the tree.

Example 4 In the preceding example, the weights $\boldsymbol{\beta}$ were chosen so that $\sigma(-14.7+5.8x)$ is an approximation to a threshold function at $x = 3$. The resulting logistic evaluation function then is a good discriminant function for identifying authors with an h-index at least 3. An alternative weight assignment $\boldsymbol{\beta} = (-1.38, 0.55, (-1.38, 0.55))$ is designed so that $\sigma(-1.38 + 0.55x)$ varies more gradually over a larger range of count values x. The resulting logistic evaluation for the TET value of Figure 3 (c) is shown in Figure 3 (d2)

3.1 Neural network perspective and weight learning

In the preceding examples we have computed the recursively defined $l^{\boldsymbol{\beta}}(\gamma)$ by an inductive bottom-up propagation of sub-values. These computations are similar to forward propagation in a neural network with sigmoid activation functions. Indeed, one can think of γ as defining a neural network structure, and $\boldsymbol{\beta}$ as defining neural network weights and biases. A given TET T with logistic evaluation parameters $\boldsymbol{\beta}$ then can be seen as a template for the construction of example-specific neural networks, and the TET formalism represents a highly flexible and expressive way to define such templates. Under this neural network perspective, then Figure 3 (d1) and (d2) show the activations of neural networks defined by the structure (c) induced by example (b), and two different weight settings $\boldsymbol{\beta}$. Note that here the inputs to the neural network are always equal to 1 at all input (leaf) nodes.

Given a supervised learning objective expressed by a differentiable loss function on the logistic evaluation values $l^{\boldsymbol{\beta}}(\gamma) = l^{\boldsymbol{\beta}}(V(T(\boldsymbol{e})))$, one can apply standard backpropagation rules to compute the gradient of the loss for a single example $\boldsymbol{e}$ with respect to $\boldsymbol{\beta}$, and learn $\boldsymbol{\beta}$ using any of the many available (stochastic) gradient descent techniques. Our experimental evaluation shows that parameter learning by backpropagation is quite effective when combined with the appropriate supervision (see results on h-index classification in Section 8.3.2).

4 Histogram Approximation

The nested multiset structures of $L^{\boldsymbol{\beta}}(V(T(\boldsymbol{e}))$ gives a detailed description of $\boldsymbol{e}$ in terms of quantitative features of its relational neighborhood, as defined by T and the parameters $\boldsymbol{\beta}$ of the logistic evaluation function. We now aim to use this description as a basis for defining distances between entities $\boldsymbol{e}, \boldsymbol{e}'$. In large and highly connected graphs, the full structure $L^{\boldsymbol{\beta}}(V(T(\boldsymbol{e}))$ will become very large, and not suitable to support fast distance computations, or to store pre-computed $L^{\boldsymbol{\beta}}(V(T(\boldsymbol{e}))$ for all $\boldsymbol{e}$.

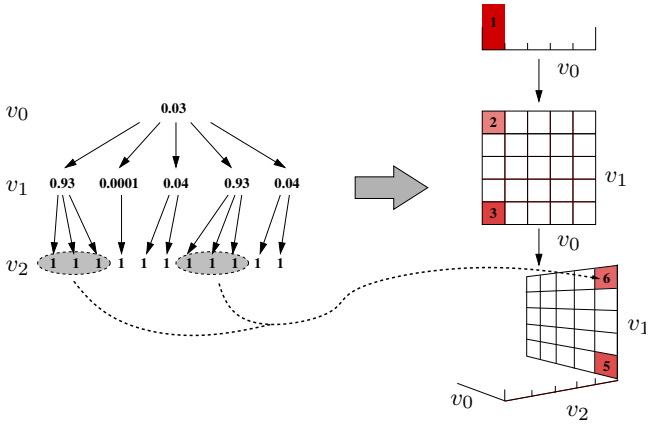


Fig. 4 Overview, continued (cf. Example 7): approximation of logistic evaluation tree by a node histogram tree with $N = 5$. A node v_i with depth d_i in the underlying TET is represented by a d_i-dimensional histogram whose dimensions correspond to the TET nodes on the path from the root to v_i. In this example, there are 6 values of the v_2 node (highlighted in the figure) whose value paths fall into the first bin for the v_0 component, and the last bin for the v_1 and v_2 components. Since all paths start with the same value (0.03) in the first component, and end with a 1 in the last component, only a single one-dimensional slice in the bottom 3-dimensional histogram is populated with nonzero counts.

We therefore introduce an approximation of $L^{\boldsymbol{\beta}}(V(T(\boldsymbol{e}))$ by a collection of multi-dimensional histograms. The approximation will be constant in size for all $\boldsymbol{e}$, and independent of the size of the data graph.

As a first step towards this approximation, we approximate the full tree structure of $L^{\boldsymbol{\beta}}(V(T(\boldsymbol{e}))$ by multisets of paths. The following definition is a bit technical, and may obscure the simple nature of what is defined. The reader may first skip forward to Examples 5 and 6 for a quick illustration of what the definition contains.

Definition 5 Let v be a node in the TET T, and $r = v_0, v_1, \ldots, v_{k-1}, v_k = v$ the path leading from the root r of T to v. Let $\gamma = V(T(\boldsymbol{e}))$ be the value of T for entities $\boldsymbol{e}$. A sequence $\gamma_0, \ldots, \gamma_k$ is a *value path for* v in γ, if

- $\gamma_0 = \gamma$, and $\gamma_i \neq f$ for all i.
- For $i < k$: if v_{i+1} is the $j(i)$th child of v_i in T, and $\gamma_i = (\mu_{i,1}, \ldots, \mu_{i,j(i)}, \ldots, \mu_{i,m_i})$, then $\gamma_{i+1} \in \mu_{i,j(i)}$ with multiplicity $k_{i+1} \geq 1$.

The *multiset of value paths for* v is the multiset that contains the value path $\gamma_0, \ldots, \gamma_k$ with multiplicity $\prod_{i=0}^{k-1} k_{i+1}$.

Example 5 For the TET in Figure 3(a) let the three nodes be v_0, v_1, v_2 indexed from top to bottom. Then the value γ shown in Figure 3(c) contains for v_2 the value path

$$\{\{t:3\}:2, \{t:2\}:2, \{t:1\}:1\},\ \{t:2\},\ t$$

with multiplicity 4 (each of these paths is induced by one of the 4 papers that cite one of a's 2 papers with citation count 2).

Let $\boldsymbol{\beta}$ be a parameter vector for the logistic evaluation function for T. A sequence of real numbers $t_1, \ldots, t_k$ is a *logistic value path for* v if there exists a value path $\gamma_0, \ldots, \gamma_k$

for v such that $t_i = l^{\boldsymbol{\beta}}(\gamma_i)$. The *multiset of logistic value paths for* v, denoted $\mathcal{M}(v)$ is the multiset that contains the logistic value path $t_1, \ldots, t_k$ with a multiplicity equal to the sum of multiplicities of value paths $\gamma_0, \ldots, \gamma_k$ that induce $t_1, \ldots, t_k$.

Example 6 Let v_0, v_1, v_2 as in Example 5. Then for the logistic evaluation tree in Figure 3 (d1):

$$\begin{aligned}\mathcal{M}(v_0) &= \{(0.03) : 1\}\\ \mathcal{M}(v_1) &= \{(0.03, 0.93) : 2, (0.03, 0.04) : 2, (0.03, 0.0001) : 1\}\\ \mathcal{M}(v_2) &= \{(0.03, 0.93, 1) : 6, (0.03, 0.04, 1) : 4,\\ &\qquad\qquad (0.03, 0.0001, 1) : 1\}\end{aligned}$$

We observe that while the value paths of a node v_{i+1} in some sense extend the value paths of its parent v_i, it is not the case that from $\mathcal{M}(v_{i+1})$ the multiset $\mathcal{M}(v_i)$ can be constructed. In the preceding example, from the occurrence of $(0.03, 0.93, 1)$ with multiplicity 6 in $\mathcal{M}(v_2)$ we can infer that $(0.03, 0.93)$ must occur in $\mathcal{M}(v_1)$, but its multiplicity in $\mathcal{M}(v_1)$ is not uniquely determined by $\mathcal{M}(v_2)$.

Values of our logistic evaluation function lie in the interval $[0, 1]$. The elements of a logistic value path multiset $\mathcal{M}(v)$ are vectors of a fixed length equal to the depth d of v in T (defining the depth of the root to be 1), and hence are elements of $[0, 1]^d$. We partition the interval $[0, 1]$ into N equal-width bins, leading to a discretization of $[0, 1]^d$ into N^d cells. This leads to an approximate representation of $\mathcal{M}(v)$ by a d-dimensional histogram, which we call the *node histogram* for v. Arranging the node histograms for all nodes of T in a tree structure isomorphic to T leads to the *node histogram tree* (NHT) as an approximation for the logistic value tree $L^{\boldsymbol{\beta}}(V(T(\boldsymbol{e})))$. The granularity N of the histogram representation is a parameter we can choose to balance accuracy with compactness of the approximation.

Example 7 Figure 4 gives a graphical illustration of the NHT constructed from the logistic evaluation tree $L^{\boldsymbol{\beta}}(V(T(a)))$ depicted in Figure 3 (d1). The counts contained in the three histograms here allow for very intuitive explanations: the top histogram just shows that the overall discriminant value for the entity a is very small, indicating that the h-index of a is less than 3. The next histogram shows that 2 of a's papers obtain a high logistic value on the sub-TET (1), indicating that they have at least 3 citations each, whereas 3 of a's papers have fewer citations. The bottom histogram associated with the leaf node $\texttt{cites}(P', P)$ is 3-dimensional. However, since all value paths end with a leaf value of 1, only the histogram slice corresponding to the maximal bin in the 3rd dimension is populated with non-zero entries. The counts in this histogram slice show: the 2 papers with at least 3 citations have a total citation count of 6, and the 3 papers with less than 3 citations have a total citation count of 5.

5 NHT Metrics

We now proceed to define a metric on NHTs. There can obviously be many different approaches for defining such a metric, and in the following we make a number of design choices. We will not be able to prove for each such choice that it is the only possible or optimal one. However, they all are supported by a few overall design objectives that we follow:

- *Based on earth mover's distance (EMD)*: the earth mover's distance (see (Rubner et al, 1998) for a standard reference) is a well-established metric on histograms representing discretized distributions of numerical quantities. Other metrics on discrete probability distributions (e.g. χ^2, Pearson or Bhattacharyya distances) or generic distances on vectors (e.g. Euclidean or cosine distance) would not take the similarity of values in nearby bins into account. EMD has proved extremely effective in tasks like image retrieval (Datta et al, 2008) requiring distances between quantized distributions. Therefore EMD is the canonical choice for measuring distances between node histograms.
- *Scale invariance*: given two NHTs H_1, H_2: if H_1', H_2' are obtained by multiplying all entries in all histograms of H_1, H_2 by a constant c, then we want to obtain $d(H_1, H_2) = d(H_1', H_2')$. In the context of count data, this seems more appropriate than location invariance (defined as invariance under *addition* of a common constant c): two authors who have written 1 and 10 papers, respectively, should be more different than two authors with 101 and 110 authored papers, but may well be considered just as different as two authors with 10 and 100 papers, respectively.
- *Few parameters*: the NHT metric should depend only on a small number of tunable parameters and work well under a simple default setting of these parameters. It is the intention that different behaviors of the final metric can be implemented by modifying the TET structure and the $\boldsymbol{\beta}$ weights of the logistic evaluation function. Adding further tunable parameters to the definition of the NHT metric would to some extent only duplicate capabilities we already have through the construction of customized NHTs.

Our construction of the NHT metric has two parts: defining a metric between node histograms of a common dimension, and combining these individual metrics into a metric on NHTs.

5.1 Node Histogram Metric

The first part is the more important one, as it is here where we have to incorporate the EMD. EMD is most naturally defined on histograms that have an equal total number of counts (usually normalized to a probability distribution over the histogram bins). For histograms with unequal total mass, the early definition given in (Rubner et al, 1998) does not lead to a proper metric. A modified definition of EMD for distributions with unequal total mass has been proposed in (Pele and Werman, 2008) (similarly also in (Ljosa et al, 2006)). The modification essentially consists of adding to the histogram with lower total count a virtual bin that has a constant distance to all other bins, and is assigned the difference of total counts in the two histograms. Then standard EMD is applied to the two now equal-sized histograms. If the constant distance of the virtual bin is at least 1/2 of the maximal distance between any of the original bins, then the result is again a proper metric (Pele and Werman, 2008). This existing approach for dealing with histograms with unequal counts is not very well suited for our purpose, since it does not lead to a scale-invariant measure, and in the case of histograms with widely different total counts (as often encountered in our applications), the differences in total counts dominate the computed distance, which then becomes less sensitive to the differences in the distributions over the histogram bins.

We therefore introduce a different approach for dealing with histograms of unequal mass that leads to a proper scale-invariant metric, and that allows us to better calibrate

the contributions to the overall distance of differences in total counts, and differences in the distributional patterns.

In the following, we use h to denote individual node histograms, and H to denote NHTs. For a histogram h we denote with $c(h)$ the sum of all cell counts in h. Let h_1, h_2 be two histograms of equal dimensions (i.e., h_1, h_2 have the same dimensionality, and the same number of bins in each dimension). Then the *relative count distance* between h_1 and h_2 is defined as:

$$d_{r\text{-}count}(h_1, h_2) := 1 - \frac{min(c(h_1), c(h_2))}{\sqrt{c(h_1) \cdot c(h_2)}}. \tag{4}$$

Equation (4) requires that $c(h_1)$ and $c(h_2)$ are both non-zero. To complete the definition, we define $d_{r\text{-}count}(h_1, h_2) = 1$ if exactly one of the $c(h_i)$ is zero, and $d_{r\text{-}count}(h_1, h_2) = 0$ if both are zero.

Proposition 1 $d_{\text{r-count}}$ *is a scale-invariant pseudo-metric with values in* $[0, 1]$.

A proof that $d_{r\text{-}count}$ is a pseudo-metric can be found in the appendix. It only is a pseudo metric on histograms, because it is defined as a function of the count values $c(h_i)$, and obviously, two different histograms can have identical counts, and hence zero $d_{r\text{-}count}$ distance. Seen as a function on the pairs of integers, $c(h_1), c(h_2)$, $d_{r\text{-}count}$ is a proper metric. The scale-invariance of $d_{r\text{-}count}$ is immediate.

Now let $\bar{h} = h/c(h)$ be the probability distribution on histogram bins obtained by normalizing h. The earth mover's distance between these normalized histograms is defined in terms of an underlying ground distance between histogram bins (Rubner et al, 1998). We take the Manhattan metric as the ground distance, because this is a very commonly used distance on histogram bins, and because it supports a computationally efficient approximation to the EMD that we will introduce in Section 5.3 below. Note however that our exact formulation is generic and can be used with any ground distance, provided an appropriate normalization is introduced. Indeed, learning the ground distance matrix from examples (Cuturi and Avis, 2014) is a promising direction for future research, as will be discussed in the conclusion of the paper. To ensure a common scale for all EMDs, regardless of the dimensionality (D) and granularity (N) of the histograms involved, we divide the raw Manhattan distance by $D(N-1)$, so that all distances between histogram bins fall into the interval $[0, 1]$. The EMD distance $d_{emd}(\bar{h}_1, \bar{h}_2)$ between two normalized histograms then is defined (and computed) as the solution of a linear program in M^2 variables, where M is the number of bins in each histogram.

Combining the count and the EMD distances via a simple mixture construction, we define:

$$d_{c\text{-}emd}(h_1, h_2) := \frac{1}{2}(d_{r\text{-}count}(h_1, h_2) + d_{emd}(\bar{h}_1, \bar{h}_2)) \tag{5}$$

This definition gives equal weight to the $d_{r\text{-}count}$ and d_{emd} components. Obviously, the mixture weights for these two components could be turned into an adjustable parameter. However, following our "few parameters" objective, for now we only consider equal weights (except that in the experimental section we will also consider the two extreme scenarios of giving all weight to either $d_{r\text{-}count}$ or d_{emd}).

Proposition 2 $d_{c\text{-}emd}$ *is a scale-invariant metric with values in* $[0, 1]$.

This proposition follows from Proposition 1, the fact that d_{emd} on normalized histograms is a proper metric, and the observation that if $h_1 \neq h_2$ then $c(h_1) \neq c(h_2)$, in which case $d_{r\text{-}count}(h_1, h_2) > 0$, or $\bar{h}_1 \neq \bar{h}_2$, in which case $d_{emd}(\bar{h}_1, \bar{h}_2) > 0$. Thus, $d_{c\text{-}emd}$ is a metric, even though its components only are pseudo-metrics.

5.2 Histogram Tree Metric

As the last step we have to combine the node histogram metrics into an overall metric on histogram trees. This will again simply be a mixture construction, but with a slightly more elaborate construction of mixture coefficients.

Consider two NHTs H_1, H_2 obtained as approximations of logistic evaluation trees $L^{\boldsymbol{\beta}}(V(T(\boldsymbol{e}_1)))$ and $L^{\boldsymbol{\beta}}(V(T(\boldsymbol{e}_2)))$ for entities $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$. Since both trees are derived from the same underlying TET T with nodes $(v_1, \ldots, v_k)$, they have identical structure, and consist of node histograms $(h_{1,1}, \ldots, h_{1,k})$ and $(h_{2,1}, \ldots, h_{2,k})$. A plain summation of per-node histogram distances $d_{c\text{-}emd}(h_{1,i}, h_{2,i})$ $(1 \leq i \leq k)$ would imply that branches with many children tend to dominate the overall distance. This is not desirable, especially given that node histograms of the children of a node to some extent duplicate the information it contains (cf. Example 6). In order to prevent this effect, we scale $d_{c\text{-}emd}(h_{1,i}, h_{2,i})$ by a factor $1/s_i$, where s_i is the number of siblings of v_i in T[1]. This leads to the following definition of a metric between NHTs, which we also denote as $d_{c\text{-}emd}$:

$$d_{c\text{-}emd}(H_1, H_2) := \sum_{i=1}^{k} \frac{1}{s_i} d_{c\text{-}emd}(h_{1,i}, h_{2,i}). \tag{6}$$

Proposition 3 *$d_{c\text{-}emd}$ is a scale-invariant metric on node histogram trees.*

5.3 Marginal EMD

The computation of $d_{c\text{-}emd}(H_1, H_2)$ with $H_i = (h_{i,1}, \ldots, h_{i,k})$ requires the computation of k EMDs. For a pair of node histograms the computation of $d_{emd}(\bar{h}_{1,j}, \bar{h}_{2,j})$ consists of a linear optimization problem in M^2 variables, with M the number of bins in the histograms. Assuming a fixed granularity of N bins in each dimension, this computation becomes exponential in the dimensionality of node histograms.

In contrast, EMD for 1-dimensional histograms w.r.t. Manhattan distance can be computed without the use of linear optimization simply by summing over the absolute difference of the cumulative distribution function (Chan et al, 2007): for normalized 1-dimensional histogram $\bar{h}_1, \bar{h}_2$ with N cells and cell values $\bar{h}_i(1), \ldots, \bar{h}_i(N)$ $(i = 1, 2)$, define the cumulative cell counts $f_i(k) := \sum_{j=1}^{k} h_i(j)$ $(k = 1, \ldots, N; i = 1, 2)$. Then

$$d_{emd}(\bar{h}_1, \bar{h}_2) = \sum_{k=1}^{N} |f_1(k) - f_2(k)|. \tag{7}$$

We can approximate $d_{emd}(\bar{h}_1, \bar{h}_2)$ by considering the 1 - dimensional marginals of the $\bar{h}_i$ as follows: for a D-dimensional normalized histogram $\bar{h}$ with N bins in each

[1] Preliminary experiments showed that a plain summation indeed achieves poor performance on TETs where different branches have very different number of children.

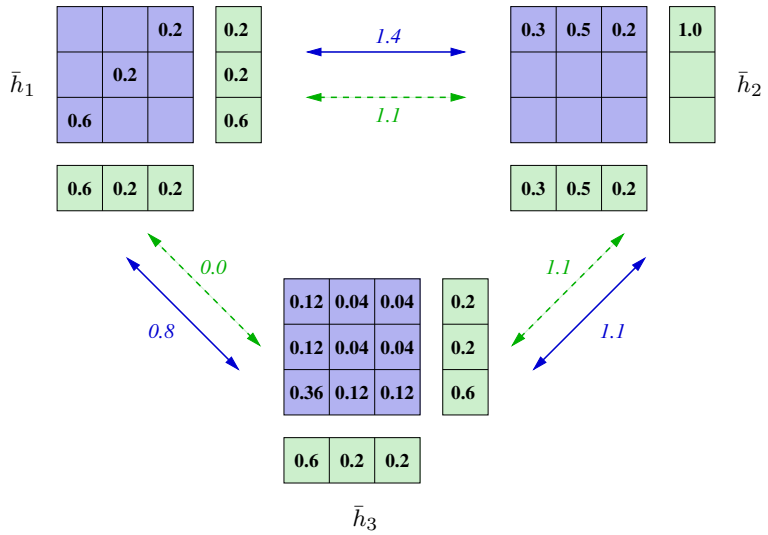


Fig. 5 2-dimensional histograms with their 1-dimensional marginals and pairwise d_{emd} (blue, solid arrows) and d_{memd} (green, dashed arrows) distances

dimension, and $1 \leq k \leq D$ let $\bar{h}^{\downarrow k}$ denote the marginal of $\bar{h}$ in the kth dimension, i.e., $\bar{h}^{\downarrow k}$ is the 1-dimensional histogram whose count in the jth bin is the sum of all counts in $\bar{h}$ over bins with index j in the kth dimension. For a two-dimensional histogram, for instance, this corresponds to computing row and column sums.

We then define the *marginal EMD distance* between $\bar{h}_1, \bar{h}_2$ as

$$d_{memd}(\bar{h}_1, \bar{h}_2) := \sum_{k=1}^{D} d_{emd}(\bar{h}_1^{\downarrow k}, \bar{h}_2^{\downarrow k}). \tag{8}$$

Proposition 4 *d_{memd} is a pseudo-metric with $d_{memd} \leq d_{emd}$.*

It should be emphasized that the inequality $d_{memd} \leq d_{emd}$ depends on our use of the Manhattan distance as the underlying metric in the EMD definition. For other metrics on histogram bins, this inequality need not hold.

Example 8 Figure 5 shows three 2-dimensional, normalized histograms $\bar{h}_1, \bar{h}_2, \bar{h}_3$, together with their 1-dimensional marginals. The pairwise distances between the histograms according to d_{emd} and d_{memd} are shown by the labels on the blue (solid), respectively green (dashed) edges. Since $\bar{h}_1$ and $\bar{h}_3$ have identical marginals, their d_{memd} is zero. In all cases $d_{memd} \leq d_{emd}$ with equality for the pair $\bar{h}_2, \bar{h}_3$.

We next investigate under which condition the equality $d_{memd} = d_{emd}$ holds. The following is a very natural definition that basically just expresses stochastic independence in histogram terms.

Definition 6 A histogram h is a *product histogram*, if the entries of h are given by the product of its marginals, i.e., for a bin with indices $\boldsymbol{i} = (i_1, \ldots, i_D)$:
$h(\boldsymbol{i}) = \prod_{k=1}^{D} h^{\downarrow k}(i_k)$.

In Example 8, histograms $\bar{h}_2$ and $\bar{h}_3$ are both product histograms.

Proposition 5 *If $\bar{h}_1, \bar{h}_2$ are product histograms, then $d_{memd}(\bar{h}_1, \bar{h}_2) = d_{emd}(\bar{h}_1, \bar{h}_2)$.*

The condition of being a product histogram may appear rather strong. However, there is a special case of product histograms that in our context is quite important, and which is exemplified by $\bar{h}_2$ in Example 8: we call a D-dimensional histogram h *concentrated in a 1-dimensional slice* if there exists $1 \leq k \leq D$, and indices $i_1, \ldots, i_{k-1}, i_{k+1}, \ldots, i_D$, such that $h(\boldsymbol{i}) \neq 0$ only for $\boldsymbol{i}$ of the form $\boldsymbol{i} = (i_1, \ldots, i_{k-1}, j, i_{k+1}, \ldots, i_D)$ for $j = 1, \ldots, N$. When h is concentrated in a 1-dimensional slice, then h is a product histogram. Thus, when comparing histograms that are concentrated in 1-dimensional slices, d_{memd} is the same as d_{emd}. Note that it is not required that the two histograms whose distance we measure are concentrated in the same 1-dimensional slice.

The reason that concentration in 1-dimensional slices is encountered quite frequently in our node histograms, is that the value paths represented by a given histogram all start with the same (root) value, and in case of leaf nodes, all end with the value 1. This implies that all 2-dimensional node histograms associated with nodes at the second level of the TET are necessarily concentrated in 1 dimension, as are those 3-dimensional node histograms at the third level that correspond to leaf nodes. Figure 4 illustrates this for our bibliometric TET. Generally, every TET that has height at most 3 only generates node histograms that are concentrated in a 1-dimensional slice.

5.4 Baseline Count Distance

In the preceding sections we have introduced a quite sophisticated metric that starts with the underlying counts-of-counts feature represented by the TET value γ, applies the customizable feature transformation through the logistic evaluation function, and then uses a combination of count based and distribution based metrics on the histogram representations of the resulting logistic value paths. In this section we define a somewhat simpler baseline metric for comparison in our experiments. The baseline is just the Euclidean distance on the marginal count values of all node histograms: using the notation from the previous sections, and in analogy to (4) and (6) define:

$$d_{b\text{-}count}(h_1, h_2) := (c(h_1) - c(h_2))^2 \tag{9}$$

$$d_{b\text{-}count}(H_1, H_2) := \sqrt{\sum_{i=1}^{k} d_{b\text{-}count}(h_{1,i}, h_{2,i})} \tag{10}$$

Note that even though for convenience we here define the $d_{b\text{-}count}$ metric via the node histograms, it is in fact independent of the distribution of counts over the bins in the histogram, and thereby does not depend on the logistic evaluation function that induces this distribution. Moreover, $d_{b\text{-}count}$ disregards the nested counts-of-counts structure represented in the underlying TET value γ, and only considers "flat" counts. Based on our bibliometric TET of Figure 2(a), the $d_{b\text{-}count}$ distance between two authors a_1, a_2 is just the Euclidean distance between the vectors $(\#p_1, \#c_1)$ and $(\#p_2, \#c_2)$, where $\#p_i$ stands for the total number of papers of author a_i, and $\#c_i$ for the total number of citations received by a_i. In contrast to the scale-invariance of $d_{c\text{-}emd}$, we have that $d_{b\text{-}count}$ is location invariant.

6 Metric Tree Retrieval

Having defined a metric over NHTs, we are now interested in efficient nearest neighbor retrieval according to that metric. There is a large body of literature on efficient exact and approximate nearest-neighbor search, mostly relying on tree-decomposition (Clarkson, 2006) or locality-sensitive-hashing (Wang et al, 2014, 2017) methods. The vast majority of hashing-based approaches is conceived for Euclidean spaces (Wang et al, 2014), as designing hash functions with locality guarantees for non-standard distances is a hard task. In this paper we rely on a simple metric tree (MT) structure based on generalized hyperplane decomposition (Uhlmann, 1991). Albeit simple, this solution proved very effective in practice, as shown by our experimental evaluation. For the type of applications that we have in mind, it will usually not be imperative that exact nearest neighbors are retrieved. For k-nearest-neighbor prediction, for example, the prediction accuracy will not usually suffer much when k very close neighbors are used, rather than the exact k nearest neighbors. In information retrieval scenarios, k very good matches for the query will often be as useful as the k best matches (noting that the underlying precise distance measure can only approximately represent user's preferences to begin with). The procedures for building and searching MTs are briefly reviewed in Appendix B.

7 Related Work

Our work is primarily related to other approaches for measuring node similarity in graphs, and information retrieval and node classification in graph data in a broader sense. It is also related to previous work on approximating EMD.

Information retrieval from graph data is often framed as the problem of finding for a given *query graph* an approximately matching sub-graph in the data graph (Tong et al, 2007; Khan et al, 2011; Mottin et al, 2014). The similarity between query graph and sub-graph sometimes is defined in terms of pairwise similarities between the nodes of the two graphs (Khan et al, 2011). In all these approaches one compares the query graph only with the sub-graph identified by the matching. Connections of nodes in this sub-graph to other nodes in the data graph not involved in the matching are not considered. Thus, a node with a low degree in the query graph can be matched with a high-degree node in the data graph, without incurring any penalty for their similarity score. This is in sharp contrast to our metric $d_{c\text{-}emd}$, which is based on a full quantitative evaluation of the nodes' neighborhoods.

When considering measures for the (dis-)similarity of nodes, one has to carefully distinguish between measures that define similarity in terms of proximity and connectivity in the graph, and measures based on local structural similarity which do not require any connectivity for two nodes to be considered similar. Our approach, as well as node similarity measures used in the context of approximate query graph matching fall into the second category, and therefore are fundamentally different from e.g. (Jeh and Widom, 2002; Sun et al, 2011; Liu et al, 2017), where similarity is induced by the existence of (short) paths connecting the nodes.

Numerous paradigms exist for defining embeddings that map nodes into a D-dimensional Euclidean space, where then similarity can be measured by standard metrics. Such paradigms can be based on probabilistic (Hoff, 2009), matrix factorization (Newman, 2006), or neural network models (Grover and Leskovec, 2016). These

approaches share with ours the two stage process of defining node feature vectors (formally, one could view our node histogram trees as vectors in Euclidean space), on which then a metric is defined. However, all these embedding approaches are again fundamentally different from ours in that similarity of feature vectors still is mostly determined by proximity in the graph. When using such feature vectors as predictors in node classification problems, one can exploit *homophily* properties, i.e., the tendency of neighboring nodes to share the same (class) attributes. However, when the node class does not exhibit homophily, then this type of feature vector will yield poor predictors.

For node classification in the absence of homophily several approaches developed in the field of *statistical relational learning* can be used (Knobbe et al, 1999; Neville et al, 2003; Assche et al, 2006; Richardson and Domingos, 2006). These frameworks typically extend classic machine learning models such as decision trees to operate on features extracted from relational neighborhoods. Most similar in spirit to our counts-of-counts features are perhaps complex aggregate features as considered in (Assche et al, 2006; Vens et al, 2014). These approaches do not define similarity measures on graph entities, and therefore can not be used for information retrieval tasks.

Similarity-based node classification is mostly treated as a graph classification problem, applied to a node's relational k-hop neighborhood or "ego graph". Thus, in (Yanardag and Vishwanathan, 2015), for instance, individual researchers in physics are classified according to their particular research fields by classifying their ego graphs in a collaboration network. This is comparable to our approach, in that a node's TET value also is derived from its relational neighborhood. However, the *graph kernels* that are used in (Yanardag and Vishwanathan, 2015) and many other approaches (Leicht et al, 2006; Shervashidze et al, 2011; Neumann et al, 2016) to measure the similarity of (neighborhood) graphs differ from TET-based node similarity measures in that they do not allow identification of a central node of interest, and similarities of nodes is only indirectly obtained as an aggregate of similarities between all the nodes or sub-structures in their neighborhood graphs.

Most similar in spirit to TET-based similarities are perhaps the *Weisfeiler-Lehman (W-L) Graph Kernels* (Shervashidze et al, 2011) which measure similarities between labeled graphs (i.e., graphs that are equipped with a single node attribute). In an iterative process, nodes are re-labeled with the multiset of labels of their neighbors. In the course of several iterations, this implicitly leads to nested multisets of labels that indirectly encode counts-of-counts statistics. However, W-L graph kernels lack a graded concept of similarity for these nested multiset structures. The W-L graph kernel only is based on testing equality of the label multisets. Our experimental evaluation will show that this limitation produces a substantial reduction of performance with respect to TET-based similarity in nearest-neighbor based prediction.

We note that while in the present paper we use TETs and logistic evaluation functions to define structure-based similarity, we have in our previous work (Jaeger et al, 2013) also exploited TETs to construct connectivity-based similarity scores: this can be done using TETs with two free variables, such that a discriminant function score $f(V(T(e_1, e_2)))$ defined on a single TET value can be directly used as a similarity measure for two entities e_1, e_2. In (Jaeger et al, 2013) this approach was used to solve the entity resolution problem on the CORA dataset, where binary TETs for pairs of bibliographic records were learned in order to predict whether the two records actually referred to the same paper. This requires mostly connectivity-based similarity, for example based on the number of shared words in the title fields of the two records.

Our approach to approximating EMD on high-dimensional histograms as a sum of marginal EMDs is related to earlier work on obtaining lower bound approximations for high-dimensional EMD problems by dimensionality reductions via certain marginalization operations (Ljosa et al, 2006; Wichterich et al, 2008). These earlier approaches were not specifically designed for EMD on histograms, and therefore did not exploit the very easy EMD computation for one-dimensional histograms, which is the cornerstone of our approach.

8 Experiments

We performed a number of experiments to investigate the usefulness of our metrics, and the quality and efficiency of nearest neighbor retrieval when facillitated by the MEMD approximation, and the MT data structure.

8.1 Data and Experimental Setup

8.1.1 Bibliometrics

Our first application domain is bibliometrics. We employed the AMiner dataset[2], which consists of a citation network comprising a total of 1,712,433 authors, 2,092,356 papers and 8,024,869 citations. From this large dataset, we extracted the 103,658 authors with h-index > 2, where the h-index (or Hirsch index) of an author is defined as the largest number h of papers having received at least h citations each.

We defined two TETs: the one shown in Figure 2 (a), representing the basic counts-of-counts citation statistics for an author, and a second one shown in Figure 8 (a), representing publication statistics of co-authors of a given author. The second TET has depth>3, so that it induces histogram trees for which d_{emd} and d_{memd} can differ.

8.1.2 IMDb

The second application domain we considered is the Internet Movie Database (IMDb).[3] We collected the version dated February 17, 2017, from which we extracted tables regarding movies, genres, actors, and business. We built a dataset of 246,285 movies (those having at least one genre attribute), and considered 9,601 actors (those appearing in more than 20 of those movies). Based on an actor's billing position in the movie's credits we constructed actor-movie relations `lead`(a, m) if actor a appears in billing position 1 or 2 of movie m, and `support`(a, m) if a's billing position is greater than 2. Furthermore, we use a generic `role`(a, m) relation when a appears in m, whether or not billing information is available. As for the business information, we assigned each movie having budget information to one of three categories: `large_budget`(m) contains movies with a budget in the top 3% within their decade, `medium_budget` contains the next 20% of most expensive movies, and `small_budget` the remaining movies.

[2] `https://aminer.org/aminernetwork`

[3] `http://www.imdb.com/`

Table 1 Overview of methods and notations

	Notation	Definition
TET weight assignment	(def)	default parameter assignment
	(man)	manual parameter assignment
	(l-cla)	parameters learned with cross-entropy loss
	(l-mse)	parameters learned with mean squared error loss
Distance Metrics	C-EMD	combined relative count and EMD distance: equation (5)
	C-MEMD	equation (5) with d_{emd} replaced by d_{memd}.
	RCOUNT	relative count distance as defined by (4)
	MEMD	marginal EMD as defined by (8)
	BCOUNT	baseline count distance as defined by (10)
	WL-GK	Weisfeiler-Lehman graph kernel
NN retrieval	...+MT	use of metric tree data structure for fast retrieval of (approximate) nearest neighbors
Non NN methods	CLA	Classificaton based on logistic evaluation function value
	REGR	Regression based on logistic evaluation function value

8.1.3 TET parameters

All our experiments require the specification of a TET, and a weight assignment for a logistic evaluation function. For the experiments in this paper all TET structures are manually defined (we have shown in (Jaeger et al, 2013) how to learn TET structures in the context of specific supervised learning problems). For the parameter setting we employ three different methods:

Default: all "bias" parameters β_0 are set to 0, and all "weight" parameters $\beta_1, \ldots, \beta_m$ are set to 1.

Manual: we set weights manually in such a way that logistic evaluations for different examples at all TET nodes are spread over the whole available interval $[0, 1]$, and not clustered at one of the saturation points 0 or 1 of the sigmoid function. In this way we obtain a more fine-grained input for the $d_{c\text{-}emd}$ and $d_{c\text{-}memd}$ metrics with counts distributed over a wide range of histogram bins, rather than being concentrated at the extreme ends of histograms. While the heuristics we use to find these manual settings could be made quite formal and even automated, we do not pursue this in greater depth for this paper, since in future work we plan to rather pursue a metric learning approach to supersede this heuristic approach.

Learned: we learn parameters according to a given supervised learning objective as described in Section 3.1. The learning objective can be a classification task, in which case we use cross entropy as the loss function, or a regression task, in which case we use mean squared error loss. We have implemented ADAM (Kingma and Ba, 2015) as the stochastic gradient descent technique, setting the maximum number of iterations to 200 and performing 20 random restarts, keeping 10% of the training set for validation.

8.1.4 Methods used

We are considering a number of different methods for prediction and retrieval. Table 1 gives an overview of methods and notation. The different weight assignments discussed in the preceding section are summarized in the first block of Table 1. The second block

summarizes the different distance metrics we have introduced. The last two of these (BCOUNT and WL-GK) are baselines that do not use TETs. Then C-MEMD+MT (man), for example, stands for the method where we use nearest-neighbor prediction based on the $d_{c\text{-}emd}$ metric induced by a TET with manual parameter setting, and using a metric tree for approximate nearest-neighbor retrieval. C-MEMD (man) is the same, but nearest neighbors are found by exhaustive search over all training examples. We also consider alternative methods where only the logistic evaluation function value $l^{\boldsymbol{\beta}}(\gamma)$ is directly used for prediction. In classification settings we then classify an example as positive if $l^{\boldsymbol{\beta}}(\gamma) > 0.5$, and in regression settings we use $l^{\boldsymbol{\beta}}(\gamma)$ directly as the estimate. Clearly, these methods are most naturally combined with parameter learning under the corresponding objective, giving us CLA (l-cla) and REGR (l-mse). However, some other combinations are also included in the experiments.

8.1.5 Experimental Setup

In our experiments we are retrieving nearest neighbors for certain *test* entities from a given *training* dataset. For the AMiner dataset, we split our whole set of 103,658 authors into 2/3 (69,106) for training, and 1/3 (34,552) for testing. For IMDb, we manually selected 50 different test actors, optimizing for diversity in the test set, while preferring better known actors in order to improve interpretability of the retrieval results. The remaining 9,551 actors constituted the training set. For all the experiments, we used $N = 5$ as the number of histogram bins. For the MT, we used $d_{max} = 12$ as the maximum tree depth, and $n_{max} = 30$ as the maximal bucket size (see Appendix B). We run experiments on a server with Intel Xeon E5-2640 v4 2.4GHz with 128GB of RAM. For earth mover's distance, we employed the FastEMD Java implementation.[4] The source code of our implementation is freely available at `https://github.com/andreapasserini/TET`.

8.2 Efficiency of MEMD and MT Approximations

In a first set of experiments, we assess the benefits of the MEMD and MT approximations for retrieval efficiency, both individually, and in combination. We considered the problem of retrieving nearest neighbors for each of the four data/TET combinations described in Section 8.1, using C-MEMD and C-EMD distances, with or without the use of the MT. Parameters were set manually for these experiments.

Table 2 reports in the first eight columns the average time (t) per test case for retrieving the nearest neighbor, and the average total number of comparisons per test case (n_c). Comparing the time measurements for C-EMD vs. C-MEMD shows a reduction of 2-4 orders of magnitude due to the use of the MEMD approximation. Comparing the time and n_c values for the versions with and without the MT data structure shows a significant gain from the use of MTs. We also investigated how balanced the constructed MTs turn out to be. For each internal node of an MT we have recorded the split ratio (size of the smaller of the two subsets into which the data is split, divided by the size of the larger subset). For each of the eight MTs constructed in this experiment, the split ratio averaged over all its nodes was between 0.4 and 0.5, showing that

[4] `https://github.com/dkoslicki/EMDeBruijn/tree/master/FastEMD/java`

Table 2 Computational cost of retrieval using 4 different methods. All times are in ms (k = × 1,000; m= × 1,000,0000).

	Retrieval								build MT		build NHT
	C-EMD		C-MEMD		C-EMD +MT		C-MEMD +MT		C-EMD	C-MEMD	
TET	t	n_c	t	n_c	t	n_c	t	n_c	t	t	t
AMiner author	21k	69k	121	69k	43	129	<1	129	483k	3k	1m
AMiner co-auth.	486k	69k	95	69k	907	123	1	148	11m	10k	16m
IMDb genre	32k	9k	56	9k	167	32	1	32	621k	4k	852k
IMDb business	10k	9k	32	9k	97	43	<1	43	203k	1k	285k

a typical split in the construction has a ratio of about 2:1. This indicates a robust tendency for fairly balanced splits, and an expected exponential reduction in the number of required comparisons from the use of MTs. Jointly, the MEMD approximation and the MT data structure lead to a retrieval time of 1ms or less for the nearest neighbor of a test example. The following two columns of Table 2 show the times needed for the construction of the MTs. Due to the many distance computations involved in the construction, building a MT based on C-MEMD is orders of magnitude faster than based on C-EMD. However, in both cases, first the TET values and NHTs for all training cases have to be computed. The time required for this is given in the last column, and turns out to be the same order of magnitude as the C-EMD-based MT construction. Therefore, the advantage of MEMD over EMD is mostly with regard to retrieval cost, and less with respect to MT construction.

In summary, we find that both the MEMD and the MT approximations lead to speedups of several orders of magnitude in nearest neighbor retrieval. However, due to the cost of the always required initial node histogram tree computations, these speedups will mostly make a difference when amortized over many retrieval operations for a fixed dataset and MT.

8.3 Effectiveness of C-MEMD Metric for Prediction and Retrieval

In this section we test the effectiveness of C-MEMD based nearest neighbor computations for a range of supervised learning and data retrieval tasks. Before we consider specific tasks, we first investigate in the following subsection how good an approximation C-MEMD+MT is of the "gold standard" retrieval method C-EMD.

8.3.1 Quality of MEMD and MT Approximation

In Section 8.2 we found that the MEMD and MT approximations lead to very significant improvements in retrieval speed. We now evaluate how well the nearest neighbors obtained under these approximations correspond to the nearest neighbors found without approximation techniques, i.e., C-EMD.

We measure the quality of the retrieved (approximate) nearest neighbors with two metrics: Normalized Discounted Cumulative Gain (NDCG) (Järvelin and Kekäläinen, 2000) to compare the rankings defined by sorted lists of nearest neighbors, and Average Ratio Error (ARE) (Arya et al, 1998) to compare the actual distances to the test entity of true and retrieved nearest neighbors. Given a ranking of k elements defined by the

Table 3 Retrieval accuracy of nearest neighbors of C-MEMD+MT with respect to C-MEMD.

TET	NDCG	ARE
AMiner author	0.72	0.08
AMiner coauthor	0.63	0.22
IMDb genre	0.51	0.14
IMDb business	0.65	0.21

sorted list of approximate nearest neighbors, DCG is defined as:

$$\mathrm{DCG}_k = \sum_{i=1}^{k} \frac{\pi_i}{\log_2(i+1)}$$

where π_i is a relevance measure of the i-th element in the ranking, which we simply define as the reciprocal of the position of this element in the true ranking. For example, if the $k = 3$ nearest neighbors retrieved by an approximate method are the 3rd, 4th and 7th element in the true ranking, then we will have $\mathrm{DCG}_3 = (1/3)/\log_2(2) + (1/4)/\log_2(3) + (1/7)/\log_2(4) = 0.562$. This metric is typically normalized to range between 0 and 1, and then called NDCG. Our 3/4/7 example would have an NDCG of 0.380. To define ARE, let d_B be the distance of the true nearest neighbor to a test example, and d_p the distance of the returned approximate nearest neighbor. Then ARE is $\frac{d_p - d_B}{d_B}$ averaged over all test examples. Parameters were again set manually for these experiments.

First we want to assess the impact of the approximations introduced by using C-MEMD instead of C-EMD, without the use of MTs. Among our four data/TET combinations, only the co-author TET for the AMiner data can produce histograms where MEMD and EMD differ, so we can only use this setting for this experiment. Since we compare two different metrics, only NDCG is a meaningful error measure. For 50 randomly selected test authors we retrieve the 3 nearest neighbors according to C-MEMD from among the full set of 69,106 training authors, and determine their positions in the "true" ranking defined by C-EMD. We obtain an average NDCG_3 value of 0.955, indicating a near perfect approximation of C-EMD via C-MEMD.

From now on we focus on C-MEMD, and next consider the loss of retrieval accuracy induced by the use of the MT data structue, i.e., we compare C-MEMD against C-MEMD+MT. Table 3 reports the NDCG and ARE error measures for the 4 different data/TET combinations. The average NDCG scores are consistently very good, indicating that a typical retrieval result is at least as good as returning the 2nd/3rd/4th true nearest neighbors (which would correspond to an NDCG of 0.563). The ARE values, in comparison, appear relatively high, corresponding in 3 out of 4 settings to an average 10-20% larger distance of the returned than the true nearest neighbor. Note, however, that the ratio error for a single example is not bounded by 1, so that the averages here can be greatly influenced by outliers.

In summary, we find that the nearest neighbors found with C-MEMD+MT approximate the C-EMD nearest neighbors very well. For the MEMD approximation this was already partly known from Proposition 5, which implies that for most of our settings the MEMD approximation is exact.

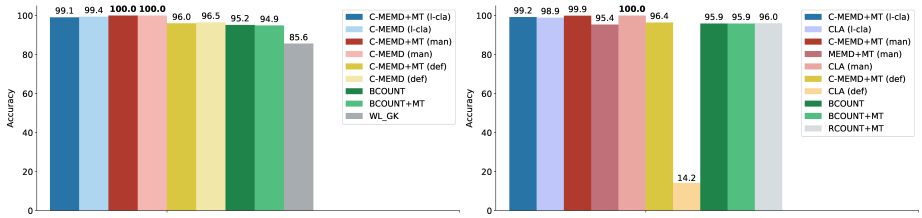


Fig. 6 Results on the bibliometrics classification task. We compare the accuracy of the considered methods on the reduced data set (left) and on the full data set (right). See Table 1 for a definition of the different methods being compared (note for viewing the figure in a B/W rendering: the top-to-bottom ordering of the methods in the key corresponds to the left-to-right ordering of the result bars).

8.3.2 Classification and regression

We next evaluate our metrics and nearest neighour retrieval in predictive tasks.

Classification. Following our earlier work (Jaeger et al, 2013) we first consider the binary classification task of predicting whether an author has an h-index greater than 7, using the citation TET of Figure 2 (a) with different methods for setting its weights. For this task the manually defined weights are set to approximate a step function going from near 0 at $x = 7$ to near 1 at $x = 8$ (instead of following the general heuristic described in Section 8.1.3 for manual weight setting), thus allowing to achieve perfect classification. We performed for the test authors a k-nearest neighbor prediction based on the $k = 10$ nearest neighbors in the training set, retrieved by C-MEMD+MT. We compare this approach to several alternatives. Some of these alternatives do not scale to our full dataset, and we therefore perform some experiments on a reduced dataset containing 10,000 authors for training, and 1,000 authors for testing. Our first comparison is against C-MEMD to assess the impact of the MT approximation on the prediction accuracy. The second comparison is against nearest-neighbor prediction, when nearest neighbors are determined based on similarity defined by the W-L graph kernel (WL-GK). The W-L graph kernel expects graphs with a single node attribute and single undirected edges between node pairs. In order to match this format, we represented the relational neighborhood information of an author entity that our citation TET exploits as an undirected graph with nodes representing ground atoms, atom types as node labels and edges connecting ground atoms according to the TET structure (see Fig 1 for examples from the IMDb domain, with node colours representing atom types). The TET based similarity $d_{c\text{-}memd}$ and the W-L kernel based similarity are thus based on the same selection of raw relational data. We used a publicly available graph kernel implementation[5] for this experiment. We set the number of iterations parameter to 1, which we found to give the best results. Our third comparison is with the baseline count metric described in Section 5.4. We ran this baseline using both exhaustive nearest neighbor search (BCOUNT) and metric tree approximation (BCOUNT+MT).

Figure 6 (left) reports classification accuracy of the methods we tested on the reduced dataset. The first relevant insight is that there is basically no difference between results of the $d_{c\text{-}memd}$ metric using exhaustive or approximate neighbor search. Comparing different weight settings, we see that manually adjusted weights give rise to

[5] `https://github.com/mahito-sugiyama/graph-kernels`

perfect classification, and that parameter learning finds nearly optimal parameters, substantially better than default ones. Competitors lag clearly behind. The WL-GK is performing the worst, most likely because its aggregation strategy fails to compute the relevant counts-of-counts statistics. The BCOUNT baselines do a reasonable job, but are substantially outperformed by the $d_{c\text{-}memd}$ metric with an appropriate weight setting (manual or learned).

Figure 6 (right) reports the results of a second set of experiments where we use the whole training/test split described in Section 8.1.5. In this case we compare C-MEMD+MT with different weight settings against three alternatives. The first alternative is given by the same count baselines used in the experiments with the reduced dataset. Results are also very similar[6], with count baselines achieving the same accuracy as in the reduced dataset. The second alternative consists of simplified versions of the metric in which only the counting component (RCOUNT+MT) or only the memd component (MEMD+MT) is used. In both cases, results are substantially worse than those achieved with the combined metric, confirming the need for considering both aspects of the similarity.

Results achieved with the direct classification model CLA and parameter settings (man) and (l-cla) are very similar to those achieved with nearest-neighbor classification. Both of these parameter settings are optimized for using the logistic evaluation function as a discriminant in this classification task. Under the (def) parameter setting the logistic evaluation always is ≥ 0.5, and therefore CLA (def) classifies all examples as positive, leading to an accuracy of only 14.2. It is noteworthy that even under this parameter setting the similarity based method C-MEMD+MT (def) performs reasonably well, indicating some robustness of the approach with respect to the parameter values.

To summarize, the main insights from the classification experiment are:

- The C-MEMD metric is expressive enough to capture under a suitable parameterization $\boldsymbol{\beta}$ the concept $h\text{-}idx > 7$ precisely.
- The classification performance of the metric is quite robust under changes of the parameterization
- Supervised parameter learning under a classification loss function here finds parameters that lead to high classification accuracy, both in the direct classification setting (CLA), and in conjunction with C-MEMD nearest neighbor prediction.
- The counts-of-counts based C-MEMD metric outperforms the flat count metric BCOUNT, where the margin of difference varies with how well the C-MEMD parameters are optimized.

Regression. We then addressed a regression task, where the goal is to exactly predict some bibliometric index of an author. We employed h, g, e, and $i10$ indices (Hirsch, 2005; Egghe, 2006; Zhang, 2009) to investigate the capability of our metric to represent relevant similarity for a broader range of prediction tasks. As before, we use the citation TET of Figure 2 (a), with different methods for setting its parameters. Note that in the l-mse weight setting, distinct parameters are learned for each index to be predicted. In the nearest neighbor based approaches, we predict each bibliometric index by the average value of the index in the 10 nearest neighbors. We compare this approach with the

[6] Note that a classification accuracy of 99.9% corresponds to F1=99.9, far higher than the one we achieved in (Jaeger et al, 2013) for the same task (on another data set) with discriminant function and nearest neighbor retrieval.

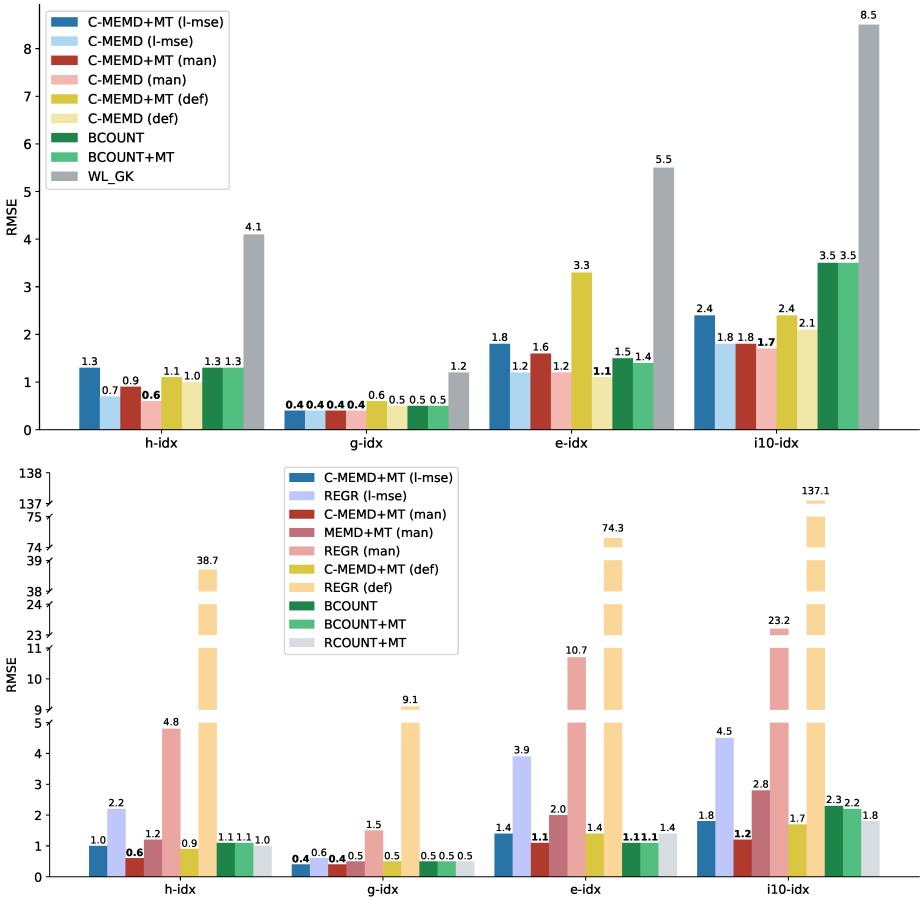


Fig. 7 Results on the bibliometrics regression task. We compare the RMSE of the considered competitors on the reduced data set (top) and on the full data set (bottom). See Table 1 for a definition of the different methods being compared (note for viewing the figure in a B/W rendering: the top-to-bottom ordering of the methods in the key corresponds to the left-to-right ordering of the result bars).

same alternatives used for the classification task, both in the reduced and full dataset settings.

Figure 7 (top) reports the root mean squared error (RMSE) of the methods we tested on the reduced dataset. A comparison of the performances of C-MEMD and C-MEMD+MT shows that for this more complex task, the MT approximation does produce some performance degradation, albeit differences are rather limited. In terms of weight settings, manual parameters again achieve the best results, and learned weights are very competitive, with almost undistinguishable results in the exact search case. In general, the metric is robust with respect to the choice of parameters, as the default weight setting is also quite competitive. In terms of alternative methods, again the WL-GK is performing the worst, and the BCOUNT baselines do a reasonable job but are outperformed by C-MEMD, especially when combined with exact search.

Figure 7 (bottom) reports the root mean squared error (RMSE) of the methods we tested on the full dataset. In this case we compare C-MEMD+MT with different weight settings againts three alternatives. The first alternative is given by the the same count baselines used in the experiments with the reduced dataset. Results are also similar, as C-MEMD+MT with manual parameters outperforms the baselines in all but the *e*-idx prediction (where they perform the same). The second alternative consists of the same simplified versions of the metric used for the classification case. As in that case, the counting and MEMD components of the metric alone give rise to substantially inferior performance than those achieved with the combined metric. With the direct regression approach REGR we only obtain competitive results with the l-mse weight setting, which substantially outperforms both manual and default parameter settings. This shows the feasibility of weight learning by backpropagation. However, even under this optimized setting of the weights, the performance of REGR is substantially worse than what we obtain from similarity-based nearest neighbor regression. To summarize, the main insights from the regression experiment are:

- C-MEMD+MT gives competitive results for all three weight setting approaches, demonstrating a certain robustness of C-MEMD based prediction under variations of the weights.
- Similarity-based nearest-neighbor prediction (C-MEMD and BCOUNT) greatly outperforms direct regression (REGR).
- The strong results of C-MEMD+MT under the manual weight setting is evidence for a potential of further improvements that could be obtained by weight optimization using a metric learning (rather than regression) approach.

8.3.3 Retrieval

In this section we present qualitative and quantitative results about the TET framework's ability to support different similarity concepts that reflect different information retrieval objectives. Compared to supervised learning tasks as considered in the previous section, retrieval tasks are much more difficult to evaluate, as there is no simple ground truth against which one can assess the results. The experiments and evaluations we present in this section, therefore, partly rely on intuitive judgement, rather than numeric performance scores.

For our retrieval experiments we use the IMDb data. Figure 8 (b) and (c) shows two TETs representing an actor's "genre profile" (b), and "business profile" (c), respectively. For the genre TET (b) we included the 20 most frequent genre labels in the dataset (the complete list can be seen in Figure 11).

For both TETs we use a manual weight setting as described in Section 8.1.3. For each of our 50 test actors (cf. Section 8.1.5) we retrieved the nearest neighbors according to the genre and the business TET. We use C-MEMD+MT for retrieval, which means that there is a certain amount of randomness in the result due to the random elements in the MT construction. Indeed, we observe some variations in the outcomes, when re-running the experiment with different random seeds.

The result for test actor Humphrey Bogart is illustrated in Figure 1 and was already discussed in Section 1. We here remark that the nearest neighbors shown in Figure 1 were retrieved using the MT data structure, and therefore cannot be guaranteed to be the precise nearest neighbors. However, as the distance values included in the figure show, the ranking we obtain is correct: Graham really is closer to Bogart than Curtis

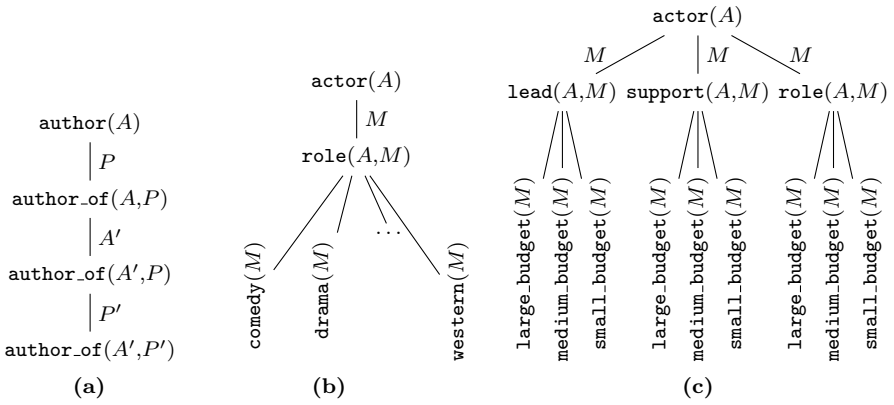


Fig. 8 TETs used for AMiner (a) and IMDb data (b),(c)

Table 4 Nearest genre and business neighbors

Test actor	NN genre	NN business
Humphrey Bogart	Eddie Graham	Tony Curtis
Stan Laurel	Billy Franey	Oliver Hardy
Joseph Stalin	Jimmy Carter	Tom Herbert
Muhammad Ali	John Kerry	Justin Ferrari
Kirk Douglas	Eli Wallach	Burt Lancaster

if looking at genres only, while the opposite holds for business similarity. Figure 10 gives the graphical representation for Stan Laurel as the test actor. Not surprisingly, Oliver Hardy here comes out as a very close neighbor both in terms of business and genre similarity, but on the latter criterion Hardy is narrowly beaten by Billy Franey (1889-1940). Franey also often appeared in leading roles, and visually there are no clear differences between either the genre, or the business feature graphs for all of these three actors.

Table 4 shows retrieved nearest genre and business neighbors for 5 selected test actors (the results for all 50 test actors can be found in the appendix). In almost all cases the nearest business and genre neighbors are distinct.

Recasting We next consider a task where an objective evaluation criterion can be defined. Named "recasting", this task is designed as follows: the IMDb website[7] lists cases where an actor turned down a role in a major movie, and also states which other actor subsequently filled that role. We can view this as a retrieval problem for a director or casting agent: find for the initially chosen actor (the *query* actor) a similar replacement actor. We denote as *target* actor the actor that ultimately played the role. The IMDb website lists a total of 20 of such query/target pairs of male actors. We omit from our experiments one pair that relates to the casting of a role in a TV series, rather than a movie, and another pair for which one of the actors is not included in our preliminary list of 9,601 actors (see Section 8.1.2).

The replacement for a query actor will usually match that query actor both with respect to the type of movies they usually perform in, as well as with respect to commercial aspects as represented by our business TET. To retrieve replacement candidates,

[7] `https://www.imdb.com/poll/pOHuVFrAcR4/`

Table 5 Results on the recasting experiment. For each movie, we report the ranking of the target actor within the list of the nearest neighbors of the query actor.

Query actor	Target actor	Movie	C-MEMD	BCOUNT
Al Pacino	Chazz Palminteri	The Usual Suspects	714	1825
Burt Reynolds	Harrison Ford	Star Wars: Episode IV	1545	2263
Daniel Day-Lewis	Viggo Mortensen	Lord of the Rings: the Fellowship of the Ring	1604	9163
Denzel Washington	Brad Pitt	Se7en	32	10
Hugh Jackman	Daniel Craig	Casino Royale	276	8020
Jack Nicholson	Al Pacino	The Godfather	2195	2573
James Caan	Jack Nicholson	One Flew Over the Cuckoo's Nest	348	547
John Travolta	Tom Hanks	Forrest Gump	17	105
Johnny Depp	Matthew Broderick	Ferris Bueller's Day Off	8467	6170
Kevin Costner	Tim Robbins	The Shawshank Redemption	256	129
Leonardo DiCaprio	Mark Wahlberg	Boogie Nights	1635	2077
Leonardo DiCaprio	Christian Bale	American Psycho	235	109
Matt Damon	Sam Worthington	Avatar	3850	3308
Sean Connery	Ian McKellen	The Hobbit: an Unexpected Journey	953	5640
Sylvester Stallone	Brad Pitt	Se7en	247	399
Tom Hanks	Tom Cruise	Jerry Maguire	2	27
Tom Selleck	Harrison Ford	Indiana Jones and the Temple of Doom	162	348
Will Smith	Keanu Reeves	The Matrix	1502	9262

we therefore construct a TET that simply joins the two TETs of Figure 8 (b) and (c) as two sub-trees under a new root with a vacuous label *true*. Since candidate replacement actors should be active in the same time period as the query actor, we need to consider only movies in a time interval immediately preceding the release date of the query movie. This can be done by adding a temporal feature to the TET as follows: when recasting a role in a movie that was released in year yyyy, we add to each node that first mentions a movie entity M just introduced by an edge label the additional constraint that M was released between yyyy-21 and yyyy-1. Thus, for example, when searching for a replacement actor for Al Pacino in "The Usual Suspects", which was released in 1995, the first branch of the business sub-TET will be modified as

$$\xrightarrow{M} \texttt{lead}(A, M), \texttt{produced\_in}(M, [1974, 1994]) \longrightarrow \ldots$$

Note that while for our experiment this means that we have to construct a customized TET for each query, this reflects a real-world scenario where a query would always be evaluated on the current database, and our varying constraints just corresponds to the fixed predicate "not older than 20 years" as defined in the current version of the database at (approximately) the time the query would be posed. We use the default parameters and the C-MEMD metric for this experiment. Nearest neighbors are retrieved by exhaustive search, without the use of metric tree data structures.

For each query actor we determine the rank of the target actor in the sorted list of nearest neighbors. The results are shown in Table 5. Considering the large number of possible replacement actors, and the fact that the eventual replacement actor (our target) may be the result of many compromises and contingencies faced in the casting process, one should not expect that the target already appears in the top 10, or so, of

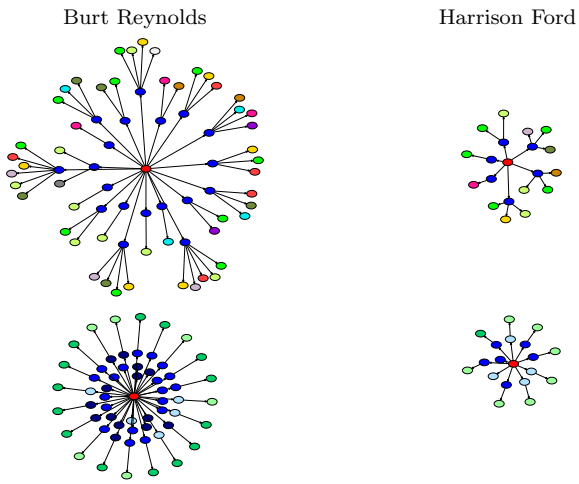


Fig. 9 Feature graphs for Burt Reynolds and Harrison Ford at production time of "Star Wars: Episode IV"

the nearest neighbor list. In our results, in 11 cases the target actor is in the top 10% of the ranking, whereas in 9 cases he is in the top 5%.

The first case where the target actor fails to make the top 10% of nearest neighbors is the Reynolds/Ford pair for "Star Wars: Episode IV". The feature graphs for these two actors provide a clear explanation for why H. Ford is not considered a very close neighbor for B. Reynolds: Figure 9 shows the genre (top) and business (bottom) feature graphs for the two actors. All graphs are constrained by the release year yyyy=1977. Obviously, at the production time of Star Wars (IV), Reynolds was a much more established actor than Ford, with a significantly larger number of movies, including numerous appearances in leading roles (darkest nodes in inner circle of Reynold's business feature graph).

We next compare the results with the C-MEMD based retrieval against the baseline BCOUNT distance as defined by the same TET. Comparing the ranks of the target actors assigned by C-MEMD vs. BCOUNT, we find that C-MEMD better ranks the target actor in 13 cases out of 18. We performed a Wilcoxon signed rank test to compare the two rankings, and the advantage of C-MEMD over BCOUNT is found to be statistically significant with $p = 0.02685$.

To summarize, we found that customizing TETs for different concepts of similarity in the same domain leads to intuitively meaningful and relevant retrieval results for the IMDb data. The newly introduced "recasting" problem supports a small-scale quantitative evaulation of the C-MEMD based similarity concept, which for this task was found to give more relevant results than the BCOUNT baseline.

9 Conclusion and Future Work

In this paper we developed a powerful class of metrics over relational data based on complex counts-of-counts statistics. The framework allows combining features such as entity attributes, attributes of the neighbors of an entity and structural properties of

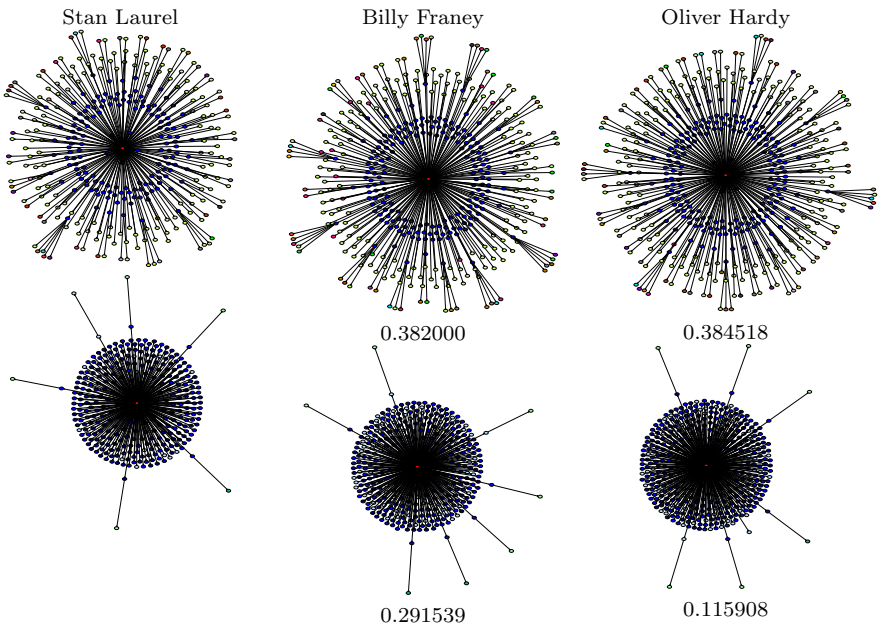


Fig. 10 Stan Laurel's nearest neighbors.

an entity relational neighborhood in a natural and flexible way. Our experimental evaluation showed how the framework allows one to seamlessly represent diverse notions of similarity and to address similarity-based prediction tasks substantially outperforming alternative solutions based on graph kernels.

Our work can be extended in a number of relevant directions. First, in the paper we only considered categorical (Boolean) attributes at the nodes. A straightforward extension is to allow numerical data on nodes and edges. A second extension concerns parameter learning. We currently either specify them manually, or learn them by stochastic gradient descent in a supervised classification/regression setting, with supervision on the output of the logistic evaluation function. In the h-index > 7 classification task, the approach learns parameters almost as accurate as the manually tuned ones. For retrieval tasks, one would like to learn the metric parameters from observed ranking preferences (Bellet et al, 2013), by e.g. adapting existing approaches for earth's mover distance learning (Wang and Guibas, 2012; Cuturi and Avis, 2014). These approaches focus on learning the ground distance matrix, i.e., the pairwise distance between bins in a (multidimensional) histogram. We plan to generalize them to also learn the parameters of the TET over which the metric is defined. Note that our experiments on bibliometric index regression suggest that this alternative form of supervision could be beneficial also in some supervised learning tasks.

Finally, our nearest-neighbor retrieval strategy relies on a simple metric tree structure based on hyperplane decomposition. The investigation of alternative search strategies, for instance by designing ad-hoc locality-sensitive hash functions, is an interesting direction for future research.

In this work we focused on the definition of a metric on NHT. It is interesting to investigate whether this metric, or variants thereof, can lead to a valid (i.e. positive

definite) kernel. Despite the importance of EMD in areas like computer vision, it is still unclear whether kernels derived from EMD (by, e.g., simply taking the negated exponential of the metric) are positive definite (Gardner et al, 2018), and there is evidence to the contrary when EMD is paired with a Euclidean ground distance (Naor and Schechtman, 2007). A sufficient and necessary condition for this to hold is that the metric is conditionally negative definite (Berg et al, 1984). It is easy to show that our MEMD metric is conditionally negative definite, also when combined with the count metric, and that $d_{c\text{-}memd}$ on node histogram trees is conditionally negative definite (see the Appendix for a formal proof). When positive definiteness cannot be ensured, one can still rely one existing solutions for dealing with indefinite kernels, most notably Krein spaces (Loosli et al, 2016; Oglic and Gaertner, 2018). We leave this investigation as an interesting avenue for future research.

References

Arya S, Mount DM, Netanyahu NS, Silverman R, Wu AY (1998) An optimal algorithm for approximate nearest neighbor searching fixed dimensions. Journal of the ACM (JACM) 45(6):891–923

Assche AV, Vens C, Blockeel H, Dzeroski S (2006) First order random forests: Learning relational classifiers with complex aggregates. Machine Learning 64:149–182

Barla A, Odone F, Verri A (2003) Histogram intersection kernel for image classification. In: Proceedings 2003 International Conference on Image Processing (Cat. No.03CH37429), vol 3, pp III–513–16 vol.2

Bellet A, Habrard A, Sebban M (2013) A survey on metric learning for feature vectors and structured data. CoRR abs/1306.6709, 1306.6709

Berg C, Christensen JP, Ressel P (1984) Harmonic analysis on semigroups: theory of positive definite and related functions, Graduate Texts in Mathematics, vol 100, 1st edn. Springer

Chan T, Esedoglu S, Ni K (2007) Histogram based segmentation using Wasserstein distances. In: International Conference on Scale Space and Variational Methods in Computer Vision, Springer, pp 697–708

Clarkson KL (2006) Nearest-neighbor searching and metric space dimensions. In: In Nearest-Neighbor Methods for Learning and Vision: Theory and Practice, MIT Press

Cuturi M, Avis D (2014) Ground metric learning. Journal of Machine Learning Research 15(1):533–564

Datta R, Joshi D, Li J, Wang JZ (2008) Image retrieval: Ideas, influences, and trends of the new age. ACM Computing Surveys 40(2):5:1–5:60

Egghe L (2006) Theory and practise of the g-index. Scientometrics 69(1):131–152

Gardner A, Duncan CA, Kanno J, Selmic RR (2018) On the definiteness of earth mover's distance and its relation to set intersection. IEEE Transactions on Cybernetics 48(11):3184–3196

Grover A, Leskovec J (2016) Node2vec: Scalable feature learning for networks. In: Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, New York, NY, USA, KDD '16, pp 855–864

Hirsch JE (2005) An index to quantify an individual's scientific research output. Proceedings of the National Academy of Sciences of the United States of America 102(46):16,569

Hoff PD (2009) Multiplicative latent factor models for description and prediction of social networks. Computational and Mathematical Organization Theory 15(4):261–272

Jaeger M, Lippi M, Passerini A, Frasconi P (2013) Type extension trees for feature construction and learning in relational domains. Artificial Intelligence 204:30–55

Järvelin K, Kekäläinen J (2000) IR evaluation methods for retrieving highly relevant documents. In: Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, pp 41–48

Jeh G, Widom J (2002) Simrank: A measure of structural-context similarity. In: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, New York, NY, USA, KDD '02, pp 538–543

Khan A, Li N, Yan X, Guan Z, Chakraborty S, Tao S (2011) Neighborhood based fast graph search in large networks. In: Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data, ACM, pp 901–912

Kingma DP, Ba J (2015) Adam: A method for stochastic optimization. In: Proceedings of the 3rd International Conference on Learning Representations (ICLR)

Knobbe AJ, Siebes A, van der Wallen D (1999) Multi-relational decision tree induction. In: Proceedings of PKDD-99, pp 378–383

Leicht EA, Holme P, Newman ME (2006) Vertex similarity in networks. Physical Review E 73(2):026,120

Liu T, Moore AW, Yang K, Gray AG (2005) An investigation of practical approximate nearest neighbor algorithms. In: Saul LK, Weiss Y, Bottou L (eds) Advances in Neural Information Processing Systems 17, MIT Press, pp 825–832

Liu Z, Zheng VW, Zhao Z, Zhu F, Chang KC, Wu M, Ying J (2017) Semantic proximity search on heterogeneous graph by proximity embedding. In: Singh SP, Markovitch S (eds) Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA., AAAI Press, pp 154–160

Ljosa V, Bhattacharya A, Singh AK (2006) Indexing spatially sensitive distance measures using multi-resolution lower bounds. In: International Conference on Extending Database Technology, Springer, pp 865–883

Loosli G, Canu S, Ong CS (2016) Learning SVM in Kreĭn Spaces. IEEE Transactions on Pattern Analysis and Machine Intelligence 38(6):1204–1216

Mottin D, Lissandrini M, Velegrakis Y, Palpanas T (2014) Exemplar queries: Give me an example of what you need. Proceedings of the VLDB Endowment 7(5):365–376

Muja M, Lowe DG (2014) Scalable nearest neighbor algorithms for high dimensional data. IEEE Transactions on Pattern Analysis and Machine Intelligence 36(11):2227–2240

Naor A, Schechtman G (2007) Planar earthmover is not in l_1. SIAM Journal on Computing 37(3):804–826

Neumann M, Garnett R, Bauckhage C, Kersting K (2016) Propagation kernels: efficient graph kernels from propagated information. Machine Learning 102(2):209–245

Neville J, Jensen D, Friedland L, Hay M (2003) Learning relational probability trees. In: Proceedings of The 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-03)

Newman ME (2006) Finding community structure in networks using the eigenvectors of matrices. Physical Review E 74(3):036,104

Oglic D, Gaertner T (2018) Learning in reproducing kernel Krein spaces. In: Dy J, Krause A (eds) Proceedings of the 35th International Conference on Machine Learning, PMLR, Proceedings of Machine Learning Research, vol 80, pp 3856–3864

Pele O, Werman M (2008) A linear time histogram metric for improved SIFT matching. In: Forsyth DA, Torr PHS, Zisserman A (eds) Computer Vision - ECCV 2008, 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part III, Springer, Lecture Notes in Computer Science, vol 5304, pp 495–508

Richards DS (1985) Positive definite symmetric functions on finite-dimensional spaces ii. Statistics & Probability Letters 3(6):325 – 329

Richardson M, Domingos P (2006) Markov logic networks. Machine Learning 62(1):107–136

Rubner Y, Tomasi C, Guibas LJ (1998) A metric for distributions with applications to image databases. In: Computer Vision, 1998. Sixth International Conference on, IEEE, pp 59–66

Schölkopf B, Smola A (2002) Learning with Kernels. The MIT Press, Cambridge, MA

Shervashidze N, Schweitzer P, van Leeuwen EJ, Mehlhorn K, Borgwardt KM (2011) Weisfeiler-Lehman graph kernels. Journal of Machine Learning Research 12:2539–2561

Sun Y, Han J, Yan X, Yu PS, Wu T (2011) Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. Proceedings of the VLDB Endowment 4(11):992–1003

Tong H, Faloutsos C, Gallagher B, Eliassi-Rad T (2007) Fast best-effort pattern matching in large attributed graphs. In: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, New York, NY, USA, KDD '07, pp 737–746

Uhlmann JK (1991) Satisfying general proximity / similarity queries with metric trees. Information Processing Letters 40(4):175 – 179

Vens C, Gassen SV, Dhaene T, Saeys Y (2014) Complex aggregates over clusters of elements. In: Davis J, Ramon J (eds) Inductive Logic Programming - 24th International Conference, ILP 2014, Nancy, France, September 14-16, 2014, Revised Selected Papers, Springer, Lecture Notes in Computer Science, vol 9046, pp 181–193

Wang F, Guibas LJ (2012) Supervised earth mover's distance learning and its computer vision applications. In: Fitzgibbon AW, Lazebnik S, Perona P, Sato Y, Schmid C (eds) Computer Vision - ECCV 2012 - 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part I, Springer, Lecture Notes in Computer Science, vol 7572, pp 442–455

Wang J, Shen HT, Song J, Ji J (2014) Hashing for similarity search: A survey. CoRR abs/1408.2927

Wang J, Zhang T, Song J, Sebe N, Shen HT (2017) A survey on learning to hash. IEEE Transactions on Pattern Analysis and Machine Intelligence PP(99):1–1

Wichterich M, Assent I, Kranen P, Seidl T (2008) Efficient emd-based similarity search in multimedia databases via flexible dimensionality reduction. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, ACM, pp 199–212

Yanardag P, Vishwanathan S (2015) Deep graph kernels. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, pp 1365–1374

Zhang CT (2009) The e-index, complementing the h-index for excess citations. PLoS One 4(5):e5429

A Proofs

Proposition 1 *$d_{\text{r-count}}$ is a scale-invariant pseudo-metric with values in $[0, 1]$.*

Proof. The minimum of two counts is a positive semi-definite kernel, called histogram intersection kernel (Barla et al, 2003). The normalization is called cosine normalization, and the result is also a kernel (Schölkopf and Smola, 2002). Let us refer to this kernel as

$$k(h_1, h_2) = \frac{min(c(h_1), c(h_2))}{\sqrt{c(h_1) \cdot c(h_2)}}.$$

A kernel induces a pseudo-metric

$$d(h_1, h_2) = \sqrt{k(h_1, h_1) + k(h_2, h_2) - 2k(h_1, h_2)}.$$

For the normalized histogram intesection kernel we have that $0 \leq k(h_1, h_2) \leq 1$ and $k(h_1, h_1) = k(h_2, h_2) = 1$, thus $d(h_1, h_2) = \sqrt{2 - 2k(h_1, h_2)}$. The count distance is obtained as $d_{r\text{-}count}(h_1, h_2) = \frac{1}{2} d(h_1, h_2)^2$, a simplified version of the distance which preserves its properties. Non-negativity and symmetry are trivially preserved. For triangle inequality $d(h_1, h_3) \leq d(h_1, h_2) + d(h_2, h_3)$ implies that $\alpha d(h_1, h_3)^2 \leq \alpha(d(h_1, h_2) + d(h_2, h_3))^2 \leq \alpha d(h_1, h_2)^2 + \alpha d(h_2, h_3)^2$ for any $\alpha > 0$. Finally, $d_{r\text{-}count}$ is a pseudo-metric because any two distinct histograms having same counts have zero distance. □

Proposition 4 *d_{memd} is a pseudo-metric with $d_{memd} \leq d_{emd}$.*

Proof. We recall and introduce the following notation: $\bar{h}_1, \bar{h}_2$ are normalized D-dimensional histograms with N bins in each dimension. Histogram cells are indexed by index vectors $\boldsymbol{i}, \boldsymbol{j}, \ldots \in N^D$. The kth component of the index vector $\boldsymbol{i}$ is denoted $\boldsymbol{i}(k)$.

For $k = 1, \ldots, D$ we have that $d^{\downarrow k}_{memd}(\bar{h}_1, \bar{h}_2) := d_{emd}(\bar{h}_1^{\downarrow k}, \bar{h}_2^{\downarrow k})$ $(k = 1, \ldots, D)$ is a pseudo-metric on the D-dimensional histograms $\bar{h}_1, \bar{h}_2$, because it is induced by the metric d_{emd} under the non-injective mapping $\bar{h} \mapsto \bar{h}^{\downarrow k}$. d_{memd} therefore is a sum of pseudo-metrics, and therefore also a pseudo-metric.

We denote by $EMD(\bar{h}_1, \bar{h}_2)$ the constrained optimization problem defining the earth mover's distance, i.e., $d_{emd}(\bar{h}_1, \bar{h}_2)$ is the cost of the optimal solution of $EMD(\bar{h}_1, \bar{h}_2)$. A feasible solution for $EMD(\bar{h}_1, \bar{h}_2)$ is a given by $\boldsymbol{f} = (f_{\boldsymbol{i},\boldsymbol{j}})_{\boldsymbol{i},\boldsymbol{j}}$, where

$$\sum_{\boldsymbol{i}} f_{\boldsymbol{i},\boldsymbol{j}} = \bar{h}_1(\boldsymbol{j}), \ \sum_{\boldsymbol{j}} f_{\boldsymbol{i},\boldsymbol{j}} = \bar{h}_2(\boldsymbol{i})$$

The *cost* of a feasible solution is

$$cost(\boldsymbol{f}) = \sum_{\boldsymbol{i},\boldsymbol{j}} f_{\boldsymbol{i},\boldsymbol{j}} d(\boldsymbol{i}, \boldsymbol{j})$$

where d is the underlying metric on histogram cells. In our case, d is the Manhattan distance. However, all we require for this proof is that d is *additive* in the sense that there exist metrics $d^{(k)}$ on $\{1, \ldots, N\}$ $(k = 1, \ldots, D)$ such that

$$d(\boldsymbol{i}, \boldsymbol{j}) = \sum_{k=1}^{D} d^{(k)}(\boldsymbol{i}(k), \boldsymbol{j}(k)).$$

In the case of Manhattan distance, $d^{(k)}(\boldsymbol{i}(k), \boldsymbol{j}(k)) = |\boldsymbol{i}(k) - \boldsymbol{j}(k)|$.

Let $\boldsymbol{f}$ be a feasible solution for $EMD(\bar{h}_1, \bar{h}_2)$. For $k = 1, \ldots, D$ we define the marginal solutions

$$f^{\downarrow k}_{i,j} := \sum_{\substack{\boldsymbol{i}:\boldsymbol{i}(k)=i \\ \boldsymbol{j}:\boldsymbol{j}(k)=j}} f_{\boldsymbol{i},\boldsymbol{j}}$$

Then $\boldsymbol{f}^{\downarrow k} = (f^{\downarrow k}_{i,j})$ is a feasible solution solution of $EMD(\bar{h}_1^{\downarrow k}, \bar{h}_2^{\downarrow k})$, and we have

$$cost(\boldsymbol{f}) \quad = \quad \sum_{\boldsymbol{i},\boldsymbol{j}} \sum_{k=1}^{D} f_{\boldsymbol{i},\boldsymbol{j}} d^{(k)}(\boldsymbol{i}(k), \boldsymbol{j}(k)) \quad = \quad \sum_{k=1}^{D} \sum_{i,j=1}^{N} f^{\downarrow k}_{i,j} d^{(k)}(i,j) \quad = \quad \sum_{k=1}^{D} cost(\boldsymbol{f}^{\downarrow k})$$

In particular, when $\boldsymbol{f}$ is a minimal cost solution of $EMD(\bar{h}_1, \bar{h}_2)$, then we have $d_{emd}(\bar{h}_1, \bar{h}_2) = cost(\boldsymbol{f})$, and

$$\sum_{k=1}^{D} cost(\boldsymbol{f}^{\downarrow k}) \geq \sum_{k=1}^{D} d_{emd}(\bar{h}_1^{\downarrow k}, \bar{h}_2^{\downarrow k}) = d_{memd}(\bar{h}_1, \bar{h}_2)$$

□

Proposition 5 *If $\bar{h}_1, \bar{h}_2$ are product histograms, then $d_{memd}(\bar{h}_1, \bar{h}_2) = d_{emd}(\bar{h}_1, \bar{h}_2)$.*

Proof. Let $\boldsymbol{f}^{(k)}$ be feasible solutions for $EMD(\bar{h}_1^{\downarrow k}, \bar{h}_2^{\downarrow k})$ $(k = 1, \ldots, D)$. Define

$$f_{\boldsymbol{i},\boldsymbol{j}} := \prod_{k=1}^{D} f^{(k)}_{\boldsymbol{i}(k),\boldsymbol{j}(k)}.$$

Then $\boldsymbol{f} = (f_{\boldsymbol{i},\boldsymbol{j}})$ is a feasible solution for $EMD(\bar{h}_1, \bar{h}_2)$:

$$\sum_{\boldsymbol{i}} f_{\boldsymbol{i},\boldsymbol{j}} \quad = \quad \sum_{\boldsymbol{i}} \prod_{k} f^{(k)}_{\boldsymbol{i}(k),\boldsymbol{j}(k)} \quad = \quad \prod_{k} \sum_{i=1}^{N} f^{(k)}_{i,\boldsymbol{j}(k)} \quad = \quad \prod_{k} \bar{h}_2^{\downarrow k}(\boldsymbol{j}(k)) \quad = \quad \bar{h}_2(\boldsymbol{j}),$$

and similarly $\sum_{\boldsymbol{j}} f_{\boldsymbol{i},\boldsymbol{j}} = \bar{h}_1(\boldsymbol{i})$. For the cost of the solutions we obtain:

$$cost(\boldsymbol{f}) = \sum_{\boldsymbol{i},\boldsymbol{j}} \left(\sum_{k} d^{(k)}(\boldsymbol{i}(k), \boldsymbol{j}(k)) \right) \prod_{k} f^{(k)}_{\boldsymbol{i}(k),\boldsymbol{j}(k)} = \sum_{k} \sum_{\boldsymbol{i},\boldsymbol{j}} d^{(k)}(\boldsymbol{i}(k), \boldsymbol{j}(k)) \prod_{k} f^{(k)}_{\boldsymbol{i}(k),\boldsymbol{j}(k)}$$

$$= \sum_{k} \sum_{i,j=1}^{N} d^{(k)}(i,j) \sum_{\substack{\boldsymbol{i}:\boldsymbol{i}(k)=i \\ \boldsymbol{j}:\boldsymbol{j}(k)=j}} \prod_{k} f^{(k)}_{\boldsymbol{i}(k),\boldsymbol{j}(k)} = \sum_{k} \sum_{i,j=1}^{N} d^{(k)}(i,j) \sum_{i,j} f^{(k)}_{i,j} = \sum_{k} cost(\boldsymbol{f}^{(k)}).$$

This implies $d_{emd}(\bar{h}_1, \bar{h}_2) \leq \sum_k d_{emd}(\bar{h}_1^{\downarrow k}, \bar{h}_2^{\downarrow k})$, which together with Proposition 4 proves the proposition. □

Proposition 6 *$d_{c\text{-}memd}$ on node histogram trees is conditionally negative definite.*

Proof. Let us recall the definition of $d_{c\text{-}memd}$ on node histogram trees and the definition of all its components:

$$d_{c\text{-}memd}(H_1, H_2) := \sum_{i=1}^{k} \frac{\gamma^{d_i}}{s_i} d_{c\text{-}memd}(h_{1,i}, h_{2,i}) \tag{11}$$

$$d_{c\text{-}memd}(h_1, h_2) := \frac{1}{2}(d_{r\text{-}count}(h_1, h_2) + d_{memd}(\bar{h}_1, \bar{h}_2)) \tag{12}$$

$$d_{memd}(\bar{h}_1, \bar{h}_2) := \sum_{k=1}^{D} d_{emd}(\bar{h}_1^{\downarrow k}, \bar{h}_2^{\downarrow k}) \tag{13}$$

$$d_{emd}(\bar{h}_1, \bar{h}_2) := \sum_{k=1}^{N} |f_1(k) - f_2(k)| \tag{14}$$

$$d_{r\text{-}count}(h_1, h_2) := 1 - \frac{min(c(h_1), c(h_2))}{\sqrt{c(h_1) \cdot c(h_2)}} \tag{15}$$

$$\tag{16}$$

Let us prove the statement in a bottom-up fashion:

Algorithm 1 Metric Tree building.

procedure MTBUILD (d_{max},n_{max},d,$data$)
  Initialize node MN
  if $d = d_{max} \vee$ SIZE($data$) $\leq n_{max}$ **then**
    MN.bucket $\leftarrow$ $data$
    return MN
  MN.z1, MN.z2 $\leftarrow$ GETRANDOMPAIR($data$)
  $data_1, data_2 \leftarrow$ SPLITDATA($data$, MN.z1, MN.z2)
  MN.left $\leftarrow$ MTBUILD(d_{max},n_{max},$d+1$,$data_1$)
  MN.right $\leftarrow$ MTBUILD(d_{max},n_{max},$d+1$,$data_2$)
  return MN

- $d_{r\text{-}count}(h_1, h_2)$ (eq. 15) is conditionally negative definite, as $\frac{min(c(h_1),c(h_2))}{\sqrt{c(h_1)\cdot c(h_2)}}$ is positive semi-definite (see proof of proposition 1), the negation of a p.s.d. function is conditionally negative definite (Berg et al, 1984), and summing a constant value does not change conditional negative definiteness.
- $d_{emd}(\bar{h}_1, \bar{h}_2)$ (eq. 14) is a Manhattan distance and thus it is conditionally negative definite (the same holds for other distances like the Euclidean one, see (Richards, 1985) for a classical proof).

It follows that $d_{c\text{-}memd}(H_1, H_2)$ is conditionally negative definite, as it is a positively weighted sum of conditionally negative definite functions, and the property is closed under summation and multiplication by positive scalar.

□

B Procedures for Metric Tree building and retrieval

In the following we briefly review the procedures for building and searching MTs, mostly following (Uhlmann, 1991).

A MT is built from a dataset of node histogram trees, by recursively splitting data until a stopping condition is met. Algorithm 1 describes the procedure for building the MT. The algorithm has two parameters, the maximal tree depth (d_{max}) and the maximal bucket size (n_{max}) and two additional arguments, the current depth (initialized at $d = 1$) and the data to be stored ($data$), represented as a set of node histogram trees, one for each entity. A MT is made of two types of nodes, internal ones and leaves. An internal node contains two entities and two branches. A leaf node contains a set of entities (the bucket). The MT construction proceeds by splitting data and recursively calling the procedure over each of the subsets, until a stopping condition is met. If the maximal tree depth is reached, or the current set to be splitted is not larger than the maximal bucket size, a leaf node is returned. If the stopping condition is not met, two entities $z1$ and $z2$ are chosen at random from the set of data (making sure they have a non-zero distance), and data are splitted according to their distances to these entities. Data that are closer to $z1$ go to the left branch, the others go to the right one, and the procedure recurses over each of the branches in turn.

Algorithm 2 Metric Tree searching.

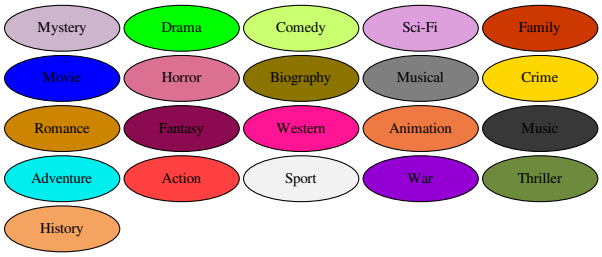
procedure MTSEARCH (MN,H,k)
  if ISLEAF(MN) **then**
    sorted = SORT(MN.bucket, H)
    return sorted[1:k]
  if DIST(H,MN.z1) $\leq$ DIST(H,MN.z2) **then**
    return MTSEARCH(MN.left,H,k)
  else
    return MTSEARCH(MN.right,H,k)

Once the MT has been built, the fastest solution for approximate k-nearest-neighbor retrieval for a query instance H amounts to traversing the tree, following at each node the branch whose corresponding entity is closer to the query one, until a leaf node is found. The entities in the bucket contained in the leaf node are then sorted according to their distance to the query entity, and the k nearest neighbors are returned. See Algorithm 2 for the pseudocode of the procedure. Notice that this is a greedy approximate solution, as exact search would require to backtrack over alternative branches, pruning a branch when it cannot contain entities closer to the query than the current k^{th} neighbor (see (Liu et al, 2005) for the details). Here we trade effectiveness for efficiency as our goal is to quickly find high quality solutions rather than discovering the actual nearest neighbors. Alternative solutions can be implemented in the latter case (Liu et al, 2005; Muja and Lowe, 2014).

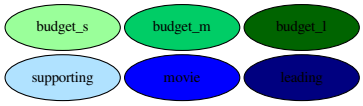
Both algorithms have as additional implicit parameter the distance function over NHTs, which can be the exact EMD-based NHT metric or its approximate version based on marginal EMD (exact for product histograms, see Proposition 5). Notice that for large databases, explicitly storing the NHT representation of each entity in the leaf buckets can be infeasible. In this case buckets only containt entity identifiers, and the corresponding NHTs are computed on-the-fly when scanning the bucket for the nearest neighbors. Standard caching solutions can be implemented to speed up this step.

C Details on actor retrieval results

Test actor	NN genre	NN business
Muhammad I Ali	John III Kerry	Justin Ferrari
Kevin I Bacon	Lance E. Nichols	Charlie Sheen
Christian Bale	Channing Tatum	Hugh I Grant
Warren I Beatty	Art I Howard	Christopher Reeve
Humphrey Bogart	Eddie I Graham	Tony I Curtis
David I Bowie	Ethan I Phillips	Adam I Baldwin
Adrien Brody	Mark I Camacho	Kevin I Kline
Steve Buscemi	Vincent I Price	Keith I David
Michael I Caine	Robert De Niro	Robert De Niro
David Carradine	Clint Howard	Rutger Hauer
Jim Carrey	Jason I Alexander	Jake Gyllenhaal
Vincent Cassel	Keith Szarabajka	Dougray Scott
James I Coburn	Ned Beatty	Louis Gossett Jr.
Robbie Coltrane	Rene Auberjonois	H.B. Warner
Sean Connery	Gene Hackman	Paul I Newman
Kirk I Douglas	Eli Wallach	Burt Lancaster
Rupert Everett	Brian Blessed	Omar Sharif
Henry Fonda	Dick I Curtis	James I Mason
John I Goodman	Christopher I Plummer	Ron I Perlman
Al I Gore	Jeroen Willems	Dwight D. Eisenhower
Dustin Hoffman	Rip Torn	Pierce Brosnan
Stan Laurel	Billy Franey	Oliver Hardy
Jude Law	Michael I Sheen	Omar Sharif
Jack Lemmon	Charles Dorety	William I Holden
John Malkovich	William H. Macy	Mickey Rourke
Marcello Mastroianni	James I Payne	Ajay Devgn
Malcolm I McDowell	Clint Howard	Martin Sheen
Alfred Molina	William H. Macy	George I Kennedy
David I Niven	Ivan F. Simpson	William I Powell
Philippe Noiret	Dominique Zardi	Pat I O'Brien
Al I Pacino	Jeremy Piven	Tom Cruise
Chazz Palminteri	Bobby Cannavale	Norman Reedus
Gregory Peck	James Seay	Christopher I Lambert
Sean I Penn	Andy I Garcia	Michael I Douglas
Anthony I Perkins	Nicholas I Campbell	George C. Scott
Joe Pesci	Stephen Marcus	Anton Yelchin
Elvis Presley	Berton Churchill	Lee I Marvin
Robert I Redford	Roscoe Ates	Michael Keaton
Keanu Reeves	Kevin I Pollak	Antonio Banderas
Geoffrey Rush	Jim I Carter	Ian I McShane
Steven Seagal	Frank Pesce	Marlon Brando
Joseph Stalin	Jimmy I Carter	Tom I Herbert
Sylvester Stallone	Nicolas Cage	Johnny Depp
Ben Stiller	Bill I Murray	Antonio Banderas
David Suchet	Danny Nucci	James I Nesbitt
John Turturro	Danny DeVito	Bruce I Dern
Lee Van Cleef	Robert I Peters	Jack Warden
Christoph Waltz	Frank I Gorshin	DemiÃ¡n Bichir
Denzel Washington	Michael V Shannon	Tom Cruise
Orson Welles	Donald Pleasence	Rod Steiger



Genre graphs



Business graphs

Fig. 11 Color keys to actor feature graphs