Instance-Driven Attachment of Semantic Annotations over Conceptual Hierarchies

Janara Christensen*
University of Washington
Seattle, Washington 98195
janara@cs.washington.edu

Marius Paşca
Google Inc.
Mountain View, California 94043
mars@google.com

Abstract

Whether automatically extracted or human generated, open-domain factual knowledge is often available in the form of semantic annotations (e.g., *composed-by*) that take one or more specific instances (e.g., *rhapsody in blue*, *george gershwin*) as their arguments. This paper introduces a method for converting flat sets of instance-level annotations into hierarchically organized, concept-level annotations, which capture not only the broad semantics of the desired arguments (e.g., 'People' rather than 'Locations'), but also the correct level of generality (e.g., 'Composers' rather than 'People', or 'Jazz Composers'). The method refrains from encoding features specific to a particular domain or annotation, to ensure immediate applicability to new, previously unseen annotations. Over a gold standard of semantic annotations and concepts that best capture their arguments, the method substantially outperforms three baselines, on average, computing concepts that are less than one step in the hierarchy away from the corresponding gold standard concepts.

1 Introduction

Background: Knowledge about the world can be thought of as semantic assertions or annotations, at two levels of granularity: instance level (e.g., *rhapsody in blue*, *tristan und isolde*, *george gershwin*, *richard wagner*) and concept level (e.g., 'Musical Compositions', 'Works of Art', 'Composers'). Instance-level annotations correspond to factual knowledge that can be found in repositories extracted automatically from text (Banko et al., 2007; Wu and Weld, 2010) or manually created within encyclopedic resources (Remy, 2002). Such facts could state, for instance, that *rhapsody in blue* was *composed-by george gershwin*, or that *tristan und isolde* was *composed-by richard wagner*. In comparison, concept-level annotations more concisely and effectively capture the underlying semantics of the annotations by identifying the concepts corresponding to the arguments, e.g., 'Musical Compositions' are *composed-by* 'Composers'.

The frequent occurrence of instances, relative to more abstract concepts, in Web documents and popular Web search queries (Barr et al., 2008; Li, 2010), is both an asset and a liability from the point of view of knowledge acquisition. On one hand, it makes instance-level annotations relatively easy to find, either from manually created resources (Remy, 2002; Bollacker et al., 2008), or extracted automatically from text (Banko et al., 2007). On the other hand, it makes concept-level annotations more difficult to acquire directly. While *"Rhapsody in Blue was composed by George Gershwin [..]"* may occur in some form within Web documents, the more abstract *"Musical compositions are composed by musicians [..]"* is unlikely to occur. A more practical approach to collecting concept-level annotations is to indirectly derive them from already plentiful instance-level annotations, effectively distilling factual knowledge into more abstract, concise and generalizable knowledge.

Contributions: This paper introduces a method for converting flat sets of specific, instance-level annotations into hierarchically organized, concept-level annotations. As illustrated in Figure 1, the resulting annotations must capture not just the broad semantics of the desired arguments (e.g., 'People' rather than 'Locations' or 'Prod-

*Contributions made during an internship at Google.

Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics, pages 503–513,
Avignon, France, April 23 - 27 2012.

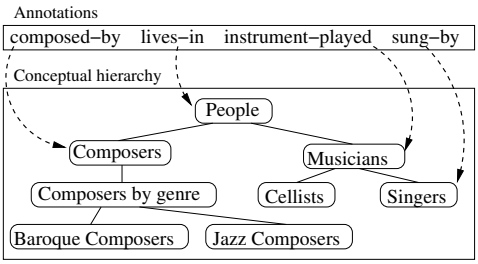

Figure 1: **Hierarchical Semantic Annotations**: The attachment of semantic annotations (e.g., *composed-by*) into a conceptual hierarchy, a portion of which is shown in the diagram, requires the identification of the correct concept at the correct level of generality (e.g., 'Composers' rather than 'Jazz Composers' or 'People', for the right argument of *composed-by*).

ucts', as the right argument of the annotation *composed-by*), but actually identify the concepts at the correct level of generality/specificity (e.g., 'Composers' rather than 'Artists' or 'Jazz Composers') in the underlying conceptual hierarchy.

To ensure portability to new, previously unseen annotations, the proposed method avoids encoding features specific to a particular domain or annotation. In particular, the use of annotations' labels (*composed-by*) as lexical features might be tempting, but would anchor the annotation model to that particular annotation. Instead, the method relies only on features that generalize across annotations. Over a gold standard of semantic annotations and concepts that best capture their arguments, the method substantially outperforms three baseline methods. On average, the method computes concepts that are less than one step in the hierarchy away from the corresponding gold standard concepts of the various annotations.

2 Hierarchical Semantic Annotations

2.1 Task Description

Data Sources: The computation of hierarchical semantic annotations relies on the following data sources:

• a target annotation r (e.g., *acted-in*) that takes M arguments;

• N annotations I=$\{<i_{1j}, \ldots, i_{Mj}>\}_{j=1}^{N}$ of r at instance level, e.g., $\{<$*leonardo dicaprio*, *inception*$>$, $<$*milla jovovich*, *fifth element*$>\}$ (in this example, M=2);

• mappings $\{i \rightarrow c\}$ from instances to concepts to which they belong, e.g., *milla jovovich* → 'American Actors', *milla jovovich* → 'People from Kiev', *milla jovovich* → 'Models';

• mappings $\{c_s \rightarrow c_g\}$ from more specific concepts to more general concepts, as encoded in a hierarchy H, e.g., 'American Actors'→'Actors', 'People from Kiev'→'People from Ukraine', 'Actors'→'Entertainers'.

Thus, the main inputs are the conceptual hierarchy H, and the instance-level annotations I. The hierarchy contains instance-to-concept mappings, as well as specific-to-general concept mappings. Via transitivity, instances (*milla jovovich*) and concepts ('American Actors') may be immediate children of more general concepts ('Actors'), or transitive descendants of more general concepts ('Entertainers'). The hierarchy is not required to be a tree; in particular, a concept may have multiple parent concepts. The instance-level annotations may be created collaboratively by human contributors, or extracted automatically from Web documents or some other data source.

Goal: Given the data sources, the goal is to determine to which concept c in the hierarchy H the arguments of the target concept-level annotation r should be attached. While the left argument of *acted-in* could attach to 'American Actors', 'People from Kiev', 'Entertainers' or 'People', it is best attached to the concept 'Actors'. The goal is to select the concept c that most appropriately generalizes across the instances. Over the set I of instance-level annotations, selecting a method for this goal can be thought of as a minimization problem. The metric to be minimized is the sum of the distances between each predicted concept c and the correct concept c_{gold}, where the distance is the number of edges between c and c_{gold} in H.

Intuitions and Challenges: Given instances such as *milla jovovich* that instantiate an argument of an annotation like *acted-in*, the conceptual hierarchy can be used to propagate the annotation upwards, from instances to their concepts, then in turn further upwards to more general concepts. The best concept would be one of the many candidate concepts reached during propagation. Intuitively, when compared to other candidate concepts, a higher proportion of the descendant instances of the best concept should instantiate (or match) the annotation. At the same time, relative to other candidate concepts, the best concept should have more descendant instances.

While the intuitions seem clear, their inclusion in a working method faces a series of practical challenges. First, the data sources may be noisy. One form of noise is missing or erroneous

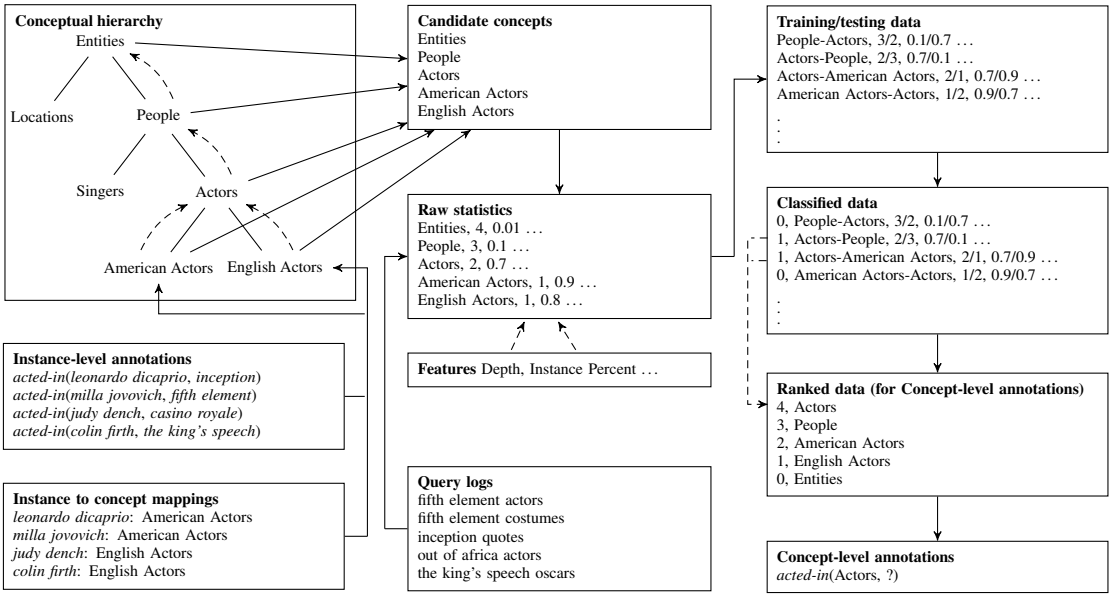

Figure 2: **Method Overview**: Inferring concept-level annotations from instance-level annotations.

instance-level annotations, which may artificially skew the distribution of matching instances towards a less than optimal region in the hierarchy. If the input annotations for *acted-in* are available almost exhaustively for all descendant instances of 'American Actors', and are available for only a few of the descendant instances of 'Belgian Actors', 'Italian Actors' etc., then the distribution over the hierarchy may incorrectly suggest that the left argument of *acted-in* is 'American Actors' rather than the more general 'Actors'. In another example, if virtually all instances that instantiate the left argument of the annotation *won-award* are mapped to the concept 'Award Winning Actors', then it would be difficult to distinguish 'Award Winning Actors' from the more general 'Actors' or 'People', as best concept to be computed for the annotation. Another type of noise is missing or erroneous edges in the hierarchy, which could artificially direct propagation towards irrelevant regions of the hierarchy, or prevent propagation from even reaching relevant regions of the hierarchy. For example, if the hierarchy incorrectly maps 'Actors' to 'Entertainment', then 'Entertainment' and its ancestor concepts incorrectly become candidate concepts during propagation for the left argument of *acted-in*. Conversely, if missing edges caused 'Actors' to not have any children in the hierarchy, then 'Actors' would not even be reached and considered as a candidate concept during propagation.

Second, to apply evidence collected from some annotations to a new annotation, the evidence must generalize across annotations. However, collected evidence or statistics may vary widely across annotations. Observing that 90% of all descendant instances of the concept 'Actors' match an annotation *acted-in* constitutes strong evidence that 'Actors' is a good concept for *acted-in*. In contrast, observing that only 0.09% of all descendant instances of the concept 'Football Teams' match *won-super-bowl* should not be as strong negative evidence as the percentage suggests.

2.2 Inferring Concept-Level Annotations

Determining Candidate Concepts: As illustrated in the left part of Figure 2, the first step towards inferring concept-level from instance-level annotations is to propagate the instances that instantiate a particular argument of the annotation, upwards in the hierarchy. Starting from the left arguments of the annotation *acted-in*, namely *leonardo dicaprio*, *milla jovovich* etc., the propagation reaches their parent concepts 'American Actors', 'English Actors', then their parent and ancestor concepts 'Actors', 'People', 'Entities' etc. The concepts reached during upward propagation become candidate concepts. In subsequent steps, the candidates are modeled, scored and ranked such that ideally the best concept is ranked at the top.

Ranking Candidate Concepts: The identifica-

tion of a ranking function is cast as a semi-supervised learning problem. Given the correct (gold) concept of an annotation, it would be tempting to employ binary classification directly, by marking the correct concept as a positive example, and all other candidate concepts as negative examples. Unfortunately, this would produce a highly imbalanced training set, with thousands of negative examples and, more importantly, with only one positive example. Another disadvantage of using binary classification directly is that it is difficult to capture the preference for concepts closer in the hierarchy to the correct concept, over concepts many edges away. Finally, the absolute values of the features that might be employed may be comparable within an annotation, but incomparable across annotations, which reduces the portability of the resulting model to new annotations.

To address the above issues, the ranking function proposed does not construct training examples from raw features collected for each individual candidate concept. Instead, it constructs training examples from pairwise comparisons of a candidate concept with another candidate concept. Concretely, a pairwise comparison is labeled as a positive example if the first concept is closer to the correct concept than the second, or as negative otherwise. The pairwise formulation has three immediate advantages. First, it accomodates the preference for concepts closer to the gold concept. Second, the pairwise formulation produces a larger, more balanced training set. Third, decisions of whether the first concept being compared is more relevant than the second are more likely to generalize across annotations, than absolute decisions of whether (and how much) a particular concept is relevant for a given annotation.

Compiling Ranking Features: The features are grouped into four categories: (A) annotation co-occurrence features, (B) concept features, (C) argument co-occurrence features, and (D) combination features, as described below.

(A) *Annotation Co-occurrence Features*: The annotation co-occurrence features emphasize how well an annotation applies to a concept. These features include (1) MATCHED INSTANCES the number of descendant instances of the concept that appear with the annotation, (2) INSTANCE PERCENT the percentage of matched instances in the concept, (3) MORE THAN THREE MATCHING INSTANCES and (4) MORE THAN TEN MATCHING INSTANCES, which indicate when the matching descendant instances might be noise.

Also in this category are features that relay information about the candidate concept's children concepts. These features include (1) MATCHED CHILDREN the number of child concepts containing at least one matching instance, (2) CHILDREN PERCENT the percentage of child concepts with at least one matching instance, (3) AVG INSTANCE PERCENT CHILDREN the average percentage of matching descendant instances of the child concepts, and (4) INSTANCE PERCENT TO INSTANCE PERCENT CHILDREN the ratio between INSTANCE PERCENT and AVERAGE INSTANCE PERCENT OF CHILDREN. The last feature is meant to capture dramatic changes in percentages when moving in the hierarchy from child concepts to the candidate concept in question.

(B) *Concept Features*: Concept features approximate the generality of the concepts: (1) NUM INSTANCES the number of descendant instances of the concept, (2) NUM CHILDREN the number of child concepts, and (3) DEPTH the distance to the concept's farthest descendant.

(C) *Argument Co-occurrence Features*: The argument co-occurrence features model the likelihood that an annotation applies to a concept by looking at co-occurrences with another argument of the same annotation. Intuitively, if a concept representing one argument has a high co-occurrence with an instance that is some other argument, a relationship more likely exists between members of the concept and the instance. For example, given *acted-in*, 'Actors' is likely to have a higher co-occurrence with *casablanca* than 'People' is. These features are generated from a set of Web queries. Therefore, the collected values are likely to be affected by different noise than that present in the original dataset. For every concept and instance pair from the arguments of a given annotation, they feature the number of times each of the tokens in the concept appears in the same query with each of the tokens in the instance, normalizing to the respective number of tokens. The procedure generates, for each candidate concept, an average co-occurrence score (AVG CO-OCCURRENCE) and a total co-occurrence score (TOTAL CO-OCCURRENCE) over all instances the concept is paired with.

(D) *Combination Features*: The last group of features are combinations of the above features: (1) DEPTH, INSTANCE PERCENT which is DEPTH multiplied by INSTANCE PERCENT, and

Concept	Distance ToCorrect	Match Inst	Total Inst	Match Child	Total Child	AvgInst PercOfChild	Depth	Avg Cooccur	Total Cooccur
People	4	36512	879423	22	29	4%	14	0.67	33506
Actors	0	29101	54420	6	10	32%	6	2.08	99971
English Actors	2	3091	5922	3	4	37%	3	2.75	28378

Labeled Concept Pair		Annotation Co-occurrence Features					Concept Features			Arg Co-occurrence Features		Combination Features	
Concept Pair	Label	Match Inst	Inst Perc	Match Child	Child Perc	AvgInst PercChild	Num Inst	Num Child	Depth	Avg Cooccur	Total Cooccur	Depth InstPerc	DepthInst PercChild
People-Actors	0	1.25	0.08	3.67	1.26	0.13	1.25	3.67	2.33	0.32	0.34	0.18	0.66
Actors-People	1	0.8	12.88	0.27	0.79	7.65	0.8	0.27	0.43	3.11	2.98	5.52	1.51
Actors-English Actors	1	9.41	1.02	2.0	0.8	0.87	9.41	2.0	2.0	0.76	3.52	2.05	4.1
English Actors-Actors	0	0.11	0.98	0.5	1.25	1.15	0.11	0.5	0.5	1.32	0.28	0.49	0.24
English Actors-People	1	0.08	12.57	0.14	0.99	8.82	0.08	0.14	0.21	4.12	0.85	2.69	0.37
People-English Actors	0	11.81	0.08	7.33	1.01	0.11	11.81	7.33	4.67	0.24	1.18	0.37	2.72

Table 1: **Training/Testing Examples**: The top table shows examples of raw statistics gathered for three candidate concepts for the left argument of the annotation *acted-in*. The second table shows the training/testing examples generated from these concepts and statistics. Each example represents a pair of concepts which is labeled positive if the first concept is closer to the correct concept than the second concept. Features shown here are the ratio between a statistic for the first concept and a statistic for the second (e.g. DEPTH for Actors-English Actors is 2 as 'Actors' has depth of 6 and 'English Actors' has depth of 3). Some features omitted due to space constraints.

(2) DEPTH, INSTANCE PERCENT, CHILDREN, which is the DEPTH multipled by the INSTANCE PERCENT multiplied by MATCHED CHILDREN. Both these features seek to balance the perceived relevance of an annotation to a candidate concept, with the generality of the candidate concept.

Generating Learning Examples: For a given annotation, the ranking features described so far are computed for each candidate concept (e.g., 'Movie Actors', 'Models', 'Actors'). However, the actual training and testing examples are generated for pairs of candidate concepts (e.g., <'Film Actors', 'Models'>, <'Film Actors', 'Actors'>, <'Models, 'Actors'>). A training example represents a comparison between two candidate concepts, and specifies which of the two is more relevant. To create training and testing examples, the values of the features of the first concept in the pair are respectively combined with the values of the features of the second concept in the pair to produce values corresponding to the entire pair.

Following classification of testing examples, concepts are ranked according to the number of other concepts which they are classified as more relevant than. Table 1 shows examples of training/testing data.

3 Experimental Setting

3.1 Data Sources

Conceptual Hierarchy: The experiments compute concept-level annotations relative to a conceptual hierarchy derived automatically from the Wikipedia (Remy, 2002) category network, as described in (Ponzetto and Navigli, 2009). The hierarchy filters out edges (e.g., from 'British Film Actors' to 'Cinema of the United Kingdom') from the Wikipedia category network that do not correspond to IsA relations. A concept in the hierarchy is a Wikipedia category (e.g., 'English Film Actors') that has zero or more Wikipedia categories as child concepts, and zero or more Wikipedia categories (e.g., 'English People by Occupation', 'British Film Actors') as parent concepts. Each concept in the hierarchy has zero or more instances, which are the Wikipedia articles listed (in Wikipedia) under the respective categories (e.g., *colin firth* is an instance of 'English Actors').

Instance-Level Annotations: The experiments exploit a set of binary instance-level annotations (e.g., *acted-in*, *composed*) among Wikipedia instances, as available in Freebase (Bollacker et al., 2008). The annotation is a Freebase property (e.g., */music/composition/composer*). Internally, the left and right arguments are Freebase topic identifiers mapped to their corresponding Wikipedia articles (e.g., */m/03f4k* mapped to the Wikipedia article on *george gershwin*). In this paper, the derived annotations and instances are displayed in a shorter, more readable form for conciseness and clarity. As features do not use the label of the annotation, labels are never used in the experiments and evaluation.

Web Search Queries: The argument co-occurrence features described above are computed over a set of around 100 million anonymized Web search queries from 2010.

3.2 Experimental Runs

The experimental runs exploit ranking features described in the previous section, employing:

• one of three learning algorithms: naive Bayes (NAIVEBAYES), maximum entropy (MAXENT), or perceptron (PERCEPTRON) (Mitchell, 1997), chosen for their scalability to larger datasets via distributed implementations.

• one of three ways of combining the values of features collected for individual candidate concepts into values of features for pairs of candidate concepts: the raw ratio of the values of the respective features of the two concepts (0 when the denominator is 0); the ratio scaled to the interval [0, 1]; or a binary value indicating which of the values is larger.

For completeness, the experiments include three additional, baseline runs. Each baseline computes scores for all candidate concepts based on the respective metric; then candidate concepts are ranked in decreasing order of their scores. The baselines metrics are:

• INSTPERCENT ranks candidate concepts by the percentage of matched instances that are descendants of the concept. It emphasizes concepts which are "proven" to belong to the annotation;

• ENTROPY ranks candidate concepts by the entropy (Shannon, 1948) of the proportion of matched descendant instances of the concept;

• AVGDEPTH ranks candidate concepts by their distances to half of the maximum hierarchy height, emphasizing a balance of generality and specificity.

3.3 Evaluation Procedure

Gold Standard of Concept-Level Annotations: A random, weighted sample of 200 annotation labels (e.g., corresponding to *composed-by*, *play-instrument*) is selected, out of the set of labels of all instance-level annotations collected from Freebase. During sampling, the weights are the counts of distinct instance-level annotations (e.g., <*rhapsody in blue*, *george gershwin*>) available for the label. The arguments of the annotation labels are then manually annotated with a gold concept, which is the category from the Wikipedia hierarchy that best captures their semantics. The manual annotation is carried out independently by two human judges, who then verify each other's work and discard inconsistencies. For example, the gold concept of the left argument of *composed-by* is annotated to be the Wikipedia category 'Musical Compositions'. In the process, some annotation labels are discarded, when (a) it is not clear what concept captures an argument (e.g., for the right argument of *function-of-building*), or (b) more than 5000 candidate concepts are available via propagation for one of the arguments, which would cause too many training or testing examples to be generated via concept pairs, and slow down the experiments. The retained 139 annotation labels, whose arguments have been labeled with their respective gold concepts, form the gold standard for the experiments. More precisely, an entry in the resulting gold standard consists of an annotation label, one of its arguments being considered (left or right), and a gold concept that best captures that argument. The set of annotation labels from the gold standard is quite diverse and covers many domains of potential interest, e.g., *has-company*('Industries', 'Companies'), *written-by*('Films', 'Screenwriters'), *member-of*('Politicians','Political Parties'), or *part-of-movement*('Artists', 'Art Movements').

Evaluation Metric: Following previous work on selectional preferences (Kozareva and Hovy, 2010; Ritter et al., 2010), each entry in the gold standard, (i.e., each argument for a given annotation) is evaluated separately. Experimental runs compute a ranked list of candidate concepts for each entry in the gold standard. In theory, a computed candidate concept is better if it is closer semantically to the gold concept. In practice, the accuracy of a ranked list of candidate concepts, relative to the gold concept of the annotation label, is measured by two scoring metrics that correspond to the mean reciprocal rank score (MRR) (Voorhees and Tice, 2000) and a modification of it (DRR) (Paşca and Alfonseca, 2009):

$$MRR = \frac{1}{N}\sum_{i=1}^{N}\max_{rank}\frac{1}{rank_i}$$

N is the number of annotations and $rank_i$ is the rank of the gold concept in the returned list for MRR. An annotation a_i receives no credit for MRR if the gold concept does not appear in the corresponding ranked list.

$$DRR = \frac{1}{N}\sum_{i=1}^{N}\max_{rank}\frac{1}{rank_i \times (1+Len)}$$

For DRR, $rank_i$ is the rank of a candidate concept in the returned list and Len is the length of

Annotation (Number of Candidate Concepts)	Examples of Instances	Top Ranked Concepts
Composers *compose* Musical Compositions (3038)	aaron copland; black sabbath	Music by Nationality; Composers; Classical Composers
Musical Compositions *composed-by* Composers (1734)	we are the champions; yorckscher marsch	Musical Compositions; Compositions by Composer; Classical Music
Foods *contain* Nutrients (1112)	acca sellowiana; lasagna	Foods; Edible Plants; Food Ingredients
Organizations *has-boardmember* People (3401)	conocophillips; spence school	Companies by Stock Exchange; Companies Listed on the NYSE; Companies
Educational Organizations *has-graduate* Alumni (4072)	air force institute of technology; deering high school	Education by Country; Schools by Country; Universities and Colleges by Country
Television Actors *guest-role* Fictional Characters (4823)	melanie griffith; patti laBelle	Television Actors by Nationality; Actors; American Actors
Musical Groups *has-member* Musicians (2287)	steroid maximus; u2	Musical Groups; Musical Groups by Genre; Musical Groups by Nationality
Record Labels *represent* Musician (920)	columbia records; vandit	Record Labels; Record Labels by Country; Record Labels by Genre
Awards *awarded-to* People (458)	academy award for best original song; erasmus prize	Film Awards; Awards; Grammy Awards
Foods *contain* **Nutrients** (177)	lycopene; glutamic acid	Carboxylic Acids ; Acids; Essential Nutrients
Architects *design* **Buildings and Structures** (4811)	20 times square; berkeley building	Buildings and Structures; Buildings and Structures by Architect; Houses by Country
People *died-from* **Causes of Death** (577)	malaria; skiing	Diseases; Infectious Diseases; Causes of Death
Art Directors *direct* **Films** (1265)	batman begins; the lion king	Films; Films by Director; Film
Episodes *guest-star* **Television Actors** (1067)	amy poehler; david caruso	Television Actors by Nationality; Actors; American Actors
Television Network *has-tv-show* **Television Series** (2492)	george of the jungle; great expectations	Television Series by Network; Television Series; Television Series by Genre
Musicians *play* **Musical Instruments** (423)	accordion; tubular bell	Musical Instruments; Musical Instruments by Nationality; Percussion Instruments
Politicians *member-of* **Political Parties** (938)	independent moralizing front; national coalition party	Political Parties; Political Parties by Country; Political Parties by Ideology

Table 2: **Concepts Computed for Gold-Standard Annotations**: Examples of entries from the gold standard and counts of candidate concepts (Wikipedia categories) reached from upward propagation of instances (Wikipedia instances). The target gold concept is shown in bold. Also shown are examples of Wikipedia instances, and the top concepts computed by the best-performing learning algorithm for the respective gold concepts.

the minimum path in the hierarchy between the concept and the gold concept. Len is minimum (0) if the candidate concept is the same as the gold standard concept. A given annotation a_i receives no credit for DRR if no path is found between the returned concepts and the gold concept.

As an illustration, for a single annotation, the right argument of *composed-by*, the ranked list of concepts returned by an experimental may be ['Symphonies by Anton Bruckner', 'Symphonies by Joseph Haydn', 'Symphonies by Gustav Mahler', 'Musical Compositions', ..], with the gold concept being 'Musical Compositions'. The length of the path between 'Symphonies by Anton Bruckner' etc. and 'Musical Compositions' is 2 (via 'Symphonies'). Therefore, the MRR score would be 0.25 (given by the fourth element of the ranked list), whereas the DRR score would be 0.33 (given by the first element of the ranked list).

MRR and DRR are computed in five-fold cross validation. Concretely, the gold standard is split into five folds such that the sets of annotation labels in each fold are disjoint. Thus, none of the annotation labels in testing appears in training. This restriction makes the evaluation more rigurous and conservative as it actually assesses the extent the models learned are applicable to new, previously unseen annotation labels. If this restriction were relaxed, the baselines would preform equivalently as they do not depend on the training data, but the learned methods would likely do better.

4 Evaluation Results

4.1 Quantitative Results

Conceptual Hierarchy: The conceptual hierarchy contains 108,810 Wikipedia categories, and its maximum depth, measured as the distance from a concept to its farthest descendant, is 16.

Candidate Concepts: On average, for the gold standard, the method propagates a given annotation from instances to 1,525 candidate concepts, from which the single best concept must be determined. The left part of Table 2 illustrates the number of candidate concepts reached during propagation for a sample of annotations.

Experimental Run	Accuracy			
	N=1		N=20	
	MRR	DRR	MRR	DRR
→ With raw-ratio features:				
NAIVEBAYES	0.021	0.180	0.054	0.222
MAXENT	0.029	0.168	0.045	0.208
PERCEPTRON	0.029	0.176	0.045	0.216
→ With scaled-ratio features:				
NAIVEBAYES	0.050	0.170	0.112	0.243
MAXENT	**0.245**	**0.456**	**0.430**	**0.513**
PERCEPTRON	0.245	0.391	0.367	0.461
→ With binary features:				
NAIVEBAYES	0.115	0.297	0.224	0.361
MAXENT	0.165	0.390	0.293	0.441
PERCEPTRON	0.180	0.332	0.330	0.429
→ For baselines:				
INSTPERCENT	0.029	0.173	0.045	0.224
ENTROPY	0.000	0.110	0.007	0.136
AVGDEPTH	0.007	0.018	0.028	0.045

Table 3: **Precision Results**: Accuracy of ranked lists of concepts (Wikipedia categories) computed by various runs, as an average over the gold standard of concept-level annotations, considering the top N candidate concepts computed for each gold standard entry.

4.2 Qualitative Results

Precision: Table 3 compares the precision of the ranked lists of candidate concepts produced by the experimental runs. The MRR and DRR scores in the table consider either at most 20 of the concepts in the ranked list computed by a given experimental run, or only the first, top ranked computed concept. Note that, in the latter case, the MRR and DRR scores are equivalent to precision@1 scores.

Several conclusions can be drawn from the results. First, as expected by definition of the scoring metrics, DRR scores are higher than the stricter MRR scores, as they give partial credit to concepts that, while not identical to the gold concepts, are still close approximations. This is particularly noticeable for the runs MAXENT and PERCEPTRON with raw-ratio features (4.6 and 4.8 times higher respectively). Second, among the baselines, INSTPERCENT is the most accurate, with the computed concepts identifying the gold concept strictly at rank 22 on average (for an MRR score 0.045), and loosely at an average of 4 steps away from the gold concept (for a DRR score of 0.224). Third, the accuracy of the learning algorithms varies with how the pairwise feature values are combined. Overall, raw-ratio feature values perform the worst, and scaled-ratio the best, with binary in-between. Fourth, the scores of the best experimental run, MAXENT with scaled-ratio features, are 0.430 (MRR) and 0.513 (DRR) over the top 20 computed concepts, and 0.245 (MRR) and 0.456 (DRR) when considering only the first concept. These scores correspond to the ranked list being less than one step away in the hierarchy. The very first computed concept exactly matches the gold concept in about one in four cases, and is slightly more than one step away from it. In comparison, the very first concept computed by the best baseline matches the gold concept in about one in 35 cases (0.029 MRR), and is about 6 steps away (0.173 DRR). The accuracies of the various learning algorithms (not shown) were also measured and correlated roughly with the MRR and DRR scores.

Discussion: The baseline runs INSTPERCENT and ENTROPY produce categories that are far too specific. For the gold annotation *composed-by*('Composers', 'Musical Compositions'), INSTPERCENT produces 'Scottish Flautists' for the left argument and 'Operas by Ernest Reyer' for the right. AVGDEPTH does not suffer from over-specification, but often produces concepts that have been reached via propagation, yet are not close to the gold concept. For *composed-by*, AVGDEPTH produces 'Film' for the left argument and 'History by Region' for the right.

4.3 Error Analysis

The right part of Table 2 provides a more detailed view into the best performing experimental run, showing actual ranked lists of concepts produced for a sample of the gold standard entries by MAXENT with scaled-ratio. A separate analysis of the results indicates that the most common cause of errors is noise in the conceptual hierarchy, in the form of *unbalanced instance-level annotations* and *missing hierarchy edges*. Unbalanced annotations are annotations where certain subtrees of the hierarchy are artificially more populated than other subtrees. For the left argument of the annotation *has-profession*, 0.05% of 'New York Politicians' are matched but 70% of 'Bushrangers' are matched. Such imbalances may be inherent to how annotations are added to Freebase: different human contributors may add new annotations to particular portions of Freebase, but miss other relevant portions.

The results are also affected by missing edges in the hierarchy. Of the more than 100K concepts in the hierarchy, 3479 are roots of subhierarchies that are mutually disconnected. Examples are 'People by Region', 'Shades of Red', and

'Members of the Parliament of Northern Ireland', all of which should have parents in the hierarchy. If a few edges are missing in a particular region of the hierarchy, the method can recover, but if so many edges are missing that a gold concept has very few descendants, then propagation can be substantially affected. In the worst case, the gold concept becomes disconnected, and thus will be missing from the set of candidate concepts compiled during propagation. For example, for the annotation *team-color*('Sports Clubs', 'Colors'), the only descendant concept of 'Colors' in the hierarchy is 'Horse Coat Colors', meaning that the gold concept 'Colors' is not reached during propagation from instances upwards in the hierarchy.

5 Related Work

Similar to the task of attaching a semantic annotation to the concept in a hierarchy that has the best level of generality is the task of finding selectional preferences for relations. Most relevant to this paper is work that seeks to find the appropriate concept in a hierarchy for an argument of a specific relation (Ribas, 1995; McCarthy, 1997; Li and Abe, 1998). Li and Abe (1998) address this problem by attempting to identify the best tree cut in a hierarchy for an argument of a given verb. They use the minimum description length principle to select a set of concepts from a hierarchy to represent the selectional preferences. This work makes several limiting assumptions including that the hierarchy is a tree, and every instance belongs to just one concept. Clark and Weir (2002) investigate the task of generalizing a single relation-concept pair. A relation is propagated up a hierarchy until a chi-square test determines the difference between the probability of the child and parent concepts to be significant where the probabilities are relation-concept frequencies. This method has no direct translation to the task discussed here; it is unclear how to choose the correct concept if instances generalize to different concepts.

In other research on selectional preferences, Pantel et al. (2007), Kozareva and Hovy (2010) and Ritter et al. (2010) focus on generating admissible arguments for relations, and Erk (2007) and Bergsma et al. (2008) investigate classifying a relation-instance pair as plausible or not.

Important to this paper is the Wikipedia category network (Remy, 2002) and work on refining it. Ponzetto and Navigli (2009) disambiguate Wikipedia categories by using WordNet synsets and use this semantic information to construct a taxonomy. The resulting taxonomy is the conceptual hierarchy used in the evaluation.

Another related area of work is the discovery of relations between concepts. Nastase and Strube (2008) use Wikipedia category names and category structure to generate a set of relations between concepts. Yan et al. (2009) discover relations between Wikipedia concepts via deep linguistic information and Web frequency information. Mohamed et al. (2011) generate candidate relations by coclustering text contexts for every pair of concepts in a hierarchy. In a sense, this area of research is complementary to that discussed in this paper. These methods induce new relations, and the proposed method can be used to find appropriate levels of generalization for the arguments of any given relation.

6 Conclusions

This paper introduces a method to convert flat sets of instance-level annotations to hierarchically organized, concept-level annotations. The method determines the appropriate concept for a given semantic annotation in three stages. First, it propagates annotations upwards in the hierarchy, forming a set of candidate concepts. Second, it classifies each candidate concept as more or less appropriate than each other candidate concept within an annotation. Third, it ranks candidate concepts by the number of other concepts relative to which it is classified as more appropriate. Because the features are comparisons between concepts within a single semantic annotation, rather than considerations of individual concepts, the method is able to generalize across annotations, and can thus be applied to new, previously unseen annotations. Experiments demonstrate that, on average, the method is able to identify the concept of a given annotation's argument within one hierarchy edge of the gold concept.

The proposed method can take advantage of existing work on open-domain information extraction. The output of such work is usually instance-level annotations, although often at surface level (non-disambiguated arguments) rather than semantic level (disambiguated arguments). After argument disambiguation (e.g., (Dredze et al., 2010)), the annotations can be used as input to determining concept-level annotations. Thus, the method has the potential to generalize any existing database of instance-level annotations to concept-level annotations.

References

Michele Banko, Michael Cafarella, Stephen Soderland, Matt Broadhead, and Oren Etzioni. 2007. Open information extraction from the Web. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI-07)*, pages 2670–2676, Hyderabad, India.

Cory Barr, Rosie Jones, and Moira Regelson. 2008. The linguistic structure of English Web-search queries. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP-08)*, pages 1021–1030, Honolulu, Hawaii.

Shane Bergsma, Dekang Lin, and Randy Goebel. 2008. Discriminative learning of selectional preference from unlabeled text. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP-08)*, pages 59–68, Honolulu, Hawaii.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 International Conference on Management of Data (SIGMOD-08)*, pages 1247–1250, Vancouver, Canada.

Stephen Clark and David Weir. 2002. Class-based probability estimation using a semantic hierarchy. *Computational Linguistics*, 28(2):187–206.

Mark Dredze, Paul McNamee, Delip Rao, Adam Gerber, and Tim Finin. 2010. Entity disambiguation for knowledge base population. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING-10)*, pages 277–285, Beijing, China.

Katrin Erk. 2007. A simple, similarity-based model for selectional preferences. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL-07)*, pages 216–223, Prague, Czech Republic.

Zornitsa Kozareva and Eduard Hovy. 2010. Learning arguments and supertypes of semantic relations using recursive patterns. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL-10)*, pages 1482–1491, Uppsala, Sweden.

Hang Li and Naoki Abe. 1998. Generalizing case frames using a thesaurus and the mdl principle. In *Proceedings of the ECAI-2000 Workshop on Ontology Learning*, pages 217–244, Berlin, Germany.

Xiao Li. 2010. Understanding the semantic structure of noun phrase queries. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL-10)*, pages 1337–1345, Uppsala, Sweden.

Diana McCarthy. 1997. Word sense disambiguation for acquisition of selectional preferences. In *Proceedings of the ACL/EACL Workshop on Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications*, pages 52–60, Madrid, Spain.

Tom Mitchell. 1997. *Machine Learing*. McGraw Hill.

Thahir Mohamed, Estevam Hruschka, and Tom Mitchell. 2011. Discovering relations between noun categories. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP-11)*, pages 1447–1455, Edinburgh, United Kingdom.

Vivi Nastase and Michael Strube. 2008. Decoding Wikipedia categories for knowledge acquisition. In *Proceedings of the 23rd National Conference on Artificial Intelligence (AAAI-08)*, pages 1219–1224, Chicago, Illinois.

M. Paşca and E. Alfonseca. 2009. Web-derived resources for Web Information Retrieval: From conceptual hierarchies to attribute hierarchies. In *Proceedings of the 32nd International Conference on Research and Development in Information Retrieval (SIGIR-09)*, pages 596–603, Boston, Massachusetts.

Patrick Pantel, Rahul Bhagat, Timothy Chklovski, and Eduard Hovy. 2007. ISP: Learning inferential selectional preferences. In *Proceedings of the Annual Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL-07)*, pages 564–571, Rochester, New York.

Simone Paolo Ponzetto and Roberto Navigli. 2009. Large-scale taxonomy mapping for restructuring and integrating Wikipedia. In *Proceedings of the 21st International Joint Conference on Artifical Intelligence (IJCAI-09)*, pages 2083–2088, Barcelona, Spain.

Melanie Remy. 2002. Wikipedia: The free encyclopedia. *Online Information Review*, 26(6):434.

Francesc Ribas. 1995. On learning more appropriate selectional restrictions. In *Proceedings of the 7th Conference of the European Chapter of the Association for Computational Linguistics (EACL-97)*, pages 112–118, Madrid, Spain.

Alan Ritter, Mausam, and Oren Etzioni. 2010. A latent dirichlet allocation method for selectional preferences. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL-10)*, pages 424–434, Uppsala, Sweden.

Claude Shannon. 1948. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423,623–656.

Ellen Voorhees and Dawn Tice. 2000. Building a question-answering test collection. In *Proceedings of the 23rd International Conference on Research and Development in Information Retrieval (SIGIR-00)*, pages 200–207, Athens, Greece.

Fei Wu and Daniel S. Weld. 2010. Open information extraction using wikipedia. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL-10)*, pages 118–127, Uppsala, Sweden.

Yulan Yan, Naoaki Okazaki, Yutaka Matsuo, Zhenglu Yang, and Mitsuru Ishizuka. 2009. Unsupervised relation extraction by mining Wikipedia texts using information from the Web. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP (ACL-IJCNLP-09)*, pages 1021–1029, Suntec, Singapore.