The Terascale Challenge

Deepak Ravichandran, Patrick Pantel, and Eduard Hovy
Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Rey, CA 90292.
{ravichan,pantel,hovy}@isi.edu

Abstract

Although vast amounts of textual data are freely available, many NLP algorithms exploit only a minute percentage. In this paper, we study the challenges of working at the terascale and survey reasons why researchers are not fully utilizing available resources. As a case study, we present a terascale algorithm for mining *is-a* relations that achieves better performance as compared to a state-of-the-art linguistically-rich method.

1 Introduction

The Natural Language Processing (NLP) community has recently seen a growth in corpus-based methods. Algorithms light in linguistic theories but rich in available training data have been successfully applied to several applications such as machine translation [15], information extraction [9], and question answering [5, 17].

In the last decade, we have seen an explosion in the amount of available digital text resources. It is estimated that the Internet contains hundreds of terabytes of text data, a sizable amount of which is in an unstructured format. State of the art search engines index more than four billion web pages. Yet, many NLP algorithms tap into only megabytes or gigabytes of this information.

In this paper, we study the challenges of working at the terascale and survey reasons why researchers are not fully utilizing available resources. We present an algorithm for extracting *is-a* relations designed for the terascale and compare it, in a preliminary study, to a state of the art method that employs deep analysis of text. We show that by simply utilizing more data on this task, we can achieve similar performance to a linguistically rich approach. *Is-a* relations are roughly characterized by the questions *What/Who is X?*. Examples of *is-a* relation are:

1. *Red* is a *color*.
2. *United States* is a *country*.
3. *Martin Luther King* was a *leader*.

In the above examples, we call *red, United States, and Martin Luther King* instances of the respective concepts *color, country and leader*.

2 Related Work

Banko and Brill [1, 2] investigated the advantages of working with very large corpora. In particular, they worked on the problem of confusion set disambiguation. It is the problem of choosing the correct use of a word from a confusion set such as {*principle, principal*}, {*then, than*}, {*to, two, too*}, and {*weather, whether*}. They empirically proved the following:

1. The learning curve is generally log-linear irrespective of the algorithm.

2. Simple and sophisticated algorithms have comparable performance with very large amounts of data. In particular, the technique of voting by using different classifiers trained on the same corpus seems to be ineffective in improving performance with large amounts of data.

3. One can achieve good performance by using supervised learning techniques and by employing active learning and sample selection.

4. Weakly supervised techniques almost seem to have little effect on performance accuracy.

Curran and Moens [7] experimented with corpus size and complexity of proximity features in building automatic thesauri. The important message to be taken home from these papers is that working with more data definitely helps.

3 Why NLP Researchers Haven't used Terabytes of Data to Date?

Statistical/Empirical techniques employed by NLP researchers to date operate on data on the order of megabytes or gigabytes. We argue that this is because of the following reasons:

1. A lot of NLP researchers have successfully made use of supervised training approaches to build several applications. Examples include POS taggers and syntactic parsers. These algorithms make use of tagged data by humans (e.g. FrameNet, Penn Tree Bank). This is a time consuming and extremely costly process.

2. Many applications such as Question Answering (QA) make use of NLP tools (e.g. syntactic parsers) which require large amounts of processing time and are not easily scalable to terabytes of data.

3. Many unsupervised algorithms (e.g. clustering algorithms) are not linear with respect to the size of the corpus. Hence, they work well only on a small corpus size and cannot be scaled to the terabyte level.

4. Terabytes of text data is not made readily available to NLP researchers by organizations like LDC (Linguistic Data Consortium). Acquiring large collections require downloading data from the Internet which is an extremely time consuming process requiring expertise in networking, distributed computing and fault tolerance.

Table 1: Projected rate of increase for various technologies.

Technology	Rate of increase
Processor Speed	100% every 1.5 years
Hard Disk Capacity	100% every year
Hard Disk Access	10% every year

5. Most of the NLP research groups do not have the necessary infrastructure (e.g. hardware, software, support staff, money) to work with such kinds of data.

4 Challenges of Working with Terabytes of data

Working on terabytes of data poses new challenges which require various engineering and algorithmic changes to the current approaches. Some of the basic challenges are:

1. Algorithms: Algorithms have to be strictly linear with respect to the size of the corpus *O(n)*. It is impossible to work with algorithms which are more than linear with the current computing power. Also, algorithms should involve only unsupervised or semi-supervised machine learning techniques. It is not trivial to hand tag data which is in the order of terabytes.

2. Storage: How would one store terabytes of data? The answer to this question is straightforward – hard disks. It is estimated that data storage capacity doubles every year[1]. A terabyte of data today costs less than $5,000. It is estimated that by the early 2010s we could buy a petabyte of data for the same cost as a terabyte costs today.

3. Data access: What is the rate at which one could access data? The data access rate from hard drives has only been growing a rate 10% a year, thus, growing an order of magnitude slower than the data storage rate. This probably means that we have to rethink the ways in which we access data. As we learnt in basic Computer Science textbooks, accessing a random location on a disk involves an overhead in terms of disk head rotation and seeking. This is a major source of delay. Disks allow roughly 200 accesses per second. So, if one reads only a few kilobytes in every disk access, it will take almost a year to read data from a 20 terabyte disk [10]. To significantly simplify our data access problems we may need to start using our disks as tapes, i.e., start using the inexpensive disks as tape drives by performing sequential access. If one reads and writes large chunks of data, data access speeds can be increased 500 times. Table 1 summarizes the differences in speeds for various technologies.

4. NLP tools: Which NLP tools could one use? One of the biggest achievements in NLP in the 1990s has been the availability of free tools to perform various tasks such as syntactic parsing, dependency parsing, discourse parsing, named-entity identification, part of speech taggers, etc. Almost all of these tools work linearly on an inter-sentence level. This is because they treat each sentence independently from other sentences. (However, in the intra-sentence level these tools may perform non-linearly

[1]This statement holds true only after 1989. Between 1960 and 1989 data storage grew only at the rate of 30%.

Table 2: Approximate processing time on a single Pentium-4 2.5 GHZ machine for a 1 Terabyte text corpus.

Tool	Processing time
POS Tagger	125 days
NP Chunker	216 days
Dependency Parser	10.2 years
Syntactic Parser	388.4 years

Table 3: Examples of *is-a* relation.

Co-occurrence-based system		Pattern-based system	
Instance	Concept	Instance	Concept
azalea	flower	American	airline
bipolar disorder	disease	Bobby Bonds	coach
Bordeaux	wine	radiation therapy	cancer treatment
Flintstones	television show	tiramisu	dessert
salmon	fish	Winona Ryder	actress

as a function of the number of words in the sentence.) We study and apply various off-the-shelf tools to data sets and estimate the amount of time taken to process a terabyte corpus. We take Brill's part of speech tagger [4], noun phrase chunker CASS [3], Lin's dependency parser Minipar [11], and Charniak's syntactic parser [6]. Results are shown in Table 2. It is very clear that terabyte-sized experiments cannot use any NLP tools in the current form.

5. Computing Power: What computer should one use? Computers have been following Moore's law: computer processing speed doubles every 18 months. An exciting development over the past years has been the availability of cluster computers to NLP researchers. Cluster computers are relatively cheaper as compared to Vector computers because they are built from cheap and mass-produced Intel processors with free Linux operating system, installed on them. Cluster computers also have a gigabit switch between them, acting like a cheap context switch. Using a cluster computer with hundreds of nodes, part of speech tagging and noun phrase chunking becomes manageable at the terascale level. However, syntactic parsers and dependency parsers are still too slow.

5 *Is-a* Relation Extraction

As a case study, we now proceed to briefly describe two models to extract of *is-a* relations: **1.** Co-occurrence model which employs linguistically-rich motivated features. **2.** Pattern-based model which employs linguistically-light features such as lexical words and POS tokens. Details of these models appear in [14]. Some examples of extracted *is-a* relations are shown in Table 3.

5.1 Co-occurrence Model

The co-occurrence model as proposed by Pantel and Ravichandran [13] employs clustering technology to extract *is-a* relations. Clustering by Committee (CBC) [12] is used to extract clusters of nouns belonging in the same semantic class. The clustering algorithm employs as features the grammatical contexts of words as output by the dependency parser Minipar [11]. As an example, the output of the clustering algorithm for the fruit sense of orange would contain the following members:

{ ... *peach, pear, apricot, strawberry, banana, mango, melon, apple, pineapple, cherry, plum, lemon, grapefruit, orange, berry, raspberry, blueberry, kiwi, ...*}

For each cluster, certain signature features are extracted which are known to signify *is-a* relations. Examples of such features include appositives (e.g. ... *Oracle*, a *company* known for its progressive employment policies, ..) and nominal subjects (e.g. ... *Apple* was a hot young *company*, with Steve Jobs in charge.). These signature features are used to extract the name of each cluster. The highest ranking name of each cluster is used as the concept for each member of the cluster. For example, the top five ranking names for a cluster containing the following elements:

{*...Curtis Joseph, John Vanbiesbrouck, Mike Richter, Tommy Salo..*}

are:

1) goalie
2) goaltender
3) goalkeeper
4) player
5) backup

The syntactical co-occurrence approach has worst case time complexity $O(n^2k)$, where n is the number of words in the corpus and k is the feature-space. Just to parse a 1 TB corpus, this approach requires approximately 10.2 years (see Table 2).

5.2 Pattern-based Model

The pattern based algorithm was specifically designed to be scalable to the terabyte level. It makes use of only POS and surface text patterns. It consists of the following steps:

1. Learn lexico-POS patterns that signify *is-a* relations using a bootstrapping approach. The following patterns are learnt from this procedure along with their underlying part of speech variations:

1. X , or Y
2. X , (a|an) Y
3. X , Y
4. Y , or X
5. X , _DT Y _(WDT|IN)
6. X is a Y
7. X , _RB known as Y
8. X (Y)
9. Y such as X
10. X, _RB called Y

11. Y like X and
12. _NN , X and other Y
13. Y , including X ,
14. Y , such as X
15. Y , especially X

2. Apply the learned patterns to a POS tagged corpus to extract *is-a* relations.

3. Apply a Maximum Entropy based machine learning filter that exploits redundancy, capitalization and other features to weed out bad relations from legitimate ones. Details of the Machine Learning filter are given in the next section.

6 Maximum Entropy Filter

In the next step, each extracted noun phrase is passed through a Machine learning filter which is a model to predict the correctness of the given *is-a* relation. In the following section, we describe the model in detail.

6.1 Model

We model a Maximum entropy model to predict the correctness of a given *is-a* relation using the following equation.

$$p(c|a,b) = \frac{exp(\sum_{m=1}^{M} \lambda_{mc} f_m(a,b))}{\sum_{c'} exp(\sum_{m=1}^{M} \lambda_{mc'} f_m(a,b))} \quad (1)$$

where,
a is the concept part of *is-a* relation.
b is the instance part of *is-a* relation.
$f_m,\ m = \{1, 2..., M\}$ are the M feature functions.
$\lambda_m,\ m = \{1, 2..., M\}$ are the corresponding M model parameters.
$c', c\ \epsilon\{false, true\}$ the decisions to be made for every instance-concept pair.

The features used to model the Eq. 1 can be classified into the following four main categories:

1. **Capitalization features**: These features check to see if certain nouns of the instance-concept begins with a capitalized letter or not. Some features are used to check if the entire instance is capitalized.

2. **Pattern-based features**: These features check to see what kind of pattern triggered this particular instance-concept pair.

3. **Lexicalized features**: These type of features checks to see if the head noun of the concept contains suffixes such as *er, or, ing, ist, man, ary, ant*. Honorific mentions such *Mr., Dr., Ms.* are also checked.

Table 4: Precision and Recall Results on True relations using Machine Learning Filter.

Sample Size	Precision	Recall
500	78%	84%

4. **Co-occurrence based features**: In this category we calculate how many times the instance-concept pair was independently observed in the corpus.

6.2 Training

We randomly sampled 1000 examples from the extracted list of *is-a* relations and asked a human to tag as *correct* or *incorrect*. We used 500 examples from the above set for training and 500 examples for testing and development.

We use Gradient Iterative Scaling algorithm (GIS) [8] to train our Maximum Entropy model implemented by YASMET [2].

6.3 Results

The results of the output of the Machine Learned filter are shown are shown in Table 4. We caution the readers that these are only the precision and recall results for the output of the Machine Learning filter. They do not measure the actual precision wherein we fuse duplicate instance-concept pairs into one output. Similarly they do not measure the actual recall of the system.

The above pattern-based algorithm runs in linear time, *O(n)*, where *n* is the size of the corpus.

7 Experiments with Corpus Size

For a pilot study, we study the task of mining *is-a* relations as a function of corpus size. For this purpose the data set is divided into different sets: 1.5 megabytes, 15 megabytes, 150 megabytes, 1.5 gigabytes, 6 gigabytes and 15 gigabytes. Three systems are evaluated:

1. Co-occurrence based system (as described in subsection 5.1).

2. Pattern-based system **without** the application of the Maximum Entropy based filter (as described in subsection 5.2)

3. Pattern-based system **with** the application of the Maximum Entropy based filter (as described in section 6).

Note that the 15GB corpus was too large to process for the Co-occurrence model. Table 5 tabulates the results. For precision calculations, we extract 50 instances (from the *is-a* list) from each system trained from different corpus size, at random. For each instance we extract the top 3 frequently occurring concepts. These are then judged

[2] YASMET – Yet Another Small Maximum Entropy Toolkit – http://www.isi.edu/∼och/YASMET/

Table 5: Approximate corpus size and instance-concept pairs extracted

Corpus	Co-occurrence		Pattern w/o filter		Pattern with filter	
Size	#	Prec.	#	Prec.	#	Prec.
1.5 MB	629	4.3%	494	38.7%	303	63.4%
15 MB	8725	14.6%	4,211	39.1%	2,914	55.6%
150 MB	93725	51.1%	40,967	40.6%	26,467	60.6%
1.5 GB	93,725	56.7%	418,949	40.4%	274,716	65.7%
6 GB	171,066	64.9%	1,398,422	46.3%	981,482	76.9%
15 GB	Too large to process		2,495,598	55.9%	1,809,579	NA
150 GB	??	??	??	??	??	??
1.5 TB	??	??	??	??	??	??

manually by a human as being *correct*, or *incorrect*. The Kappa statistic [16] measures the agreements between a set of judges assessments correcting for chance agreements:

$$K = \frac{P(A)-P(E)}{1-P(E)} \tag{2}$$

where, *P(A)* is the probability of agreement between the judges and *P(E)* is the probability that the judges agree by chance on an assessment. An experiment with $K = 0.8$ is generally viewed as reliable and $0.67 < K < 0.8$ allows tentative conclusions.

Results for System 1 (Co-occurrence) and System 2 (Pattern based without filter) were evaluated with two human judges. The reported Kappa statistics agreement has a score greater than $k = 0.75$. However, the evaluation of System 3 is preliminary and was performed by only one judge.

The graph in Figure 1 shows that the relation between the number of extracted instance-concept pairs and the corpus size is linear for both pattern-based systems. However, for the co-occurrence based system, the same relation is sub-linear. Note that the x-axis (corpus-size) of the graph is on a log scale while the y-axis (extracted relation-pairs) is on a linear scale.

Figure 2 shows the relationship between the precision of the extracted relations and the corpus size. It is clear that the precision of each system increases with more data. We suspect that the precision curve is log-linear. However, only working on a larger corpus size will prove this point. For small datasets (below 150MB), the pattern-based (without filter) method achieves higher precision compared to the co-occurrence method since the latter requires a certain critical mass of statistics before it can extract useful class signatures. On the other hand, the pattern-based approach has relatively constant precision since most of the *is-a* relations selected by it are fired by a single pattern. Once the co-occurrence system reaches its critical mass (at around 150MB), it generates much more precise *is-a* relations. The pattern-based method with filter shows a lot of promise. However, we again wish to caution the reader that the evaluations for the pattern-based system with filter was performed using only one human judge and hence the results are preliminary.

On the 6 GB corpus, the co-occurrence approach took approximately 47 single Pentium-4 2.5 GHZ processor days to complete, whereas it took the pattern-based ap-

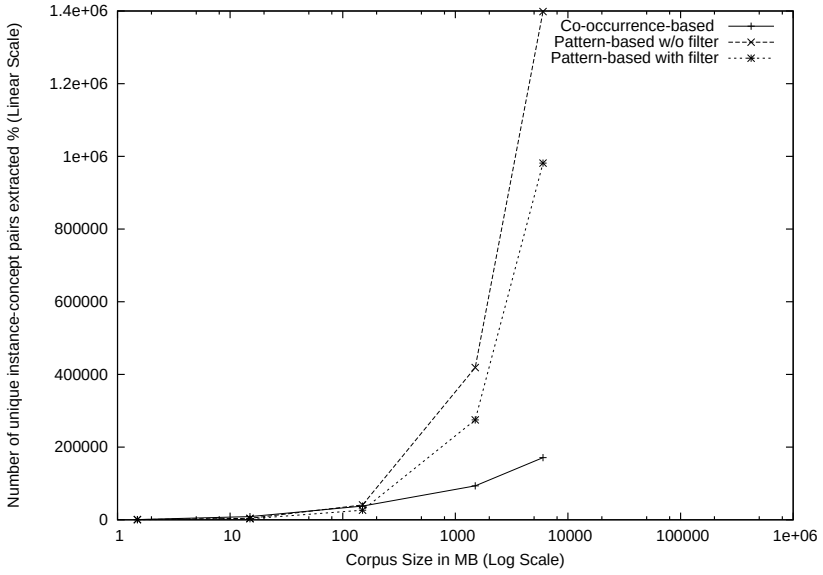

Figure 1: Graph showing the number of unique instance-concept pair extracted as a function of corpus size.

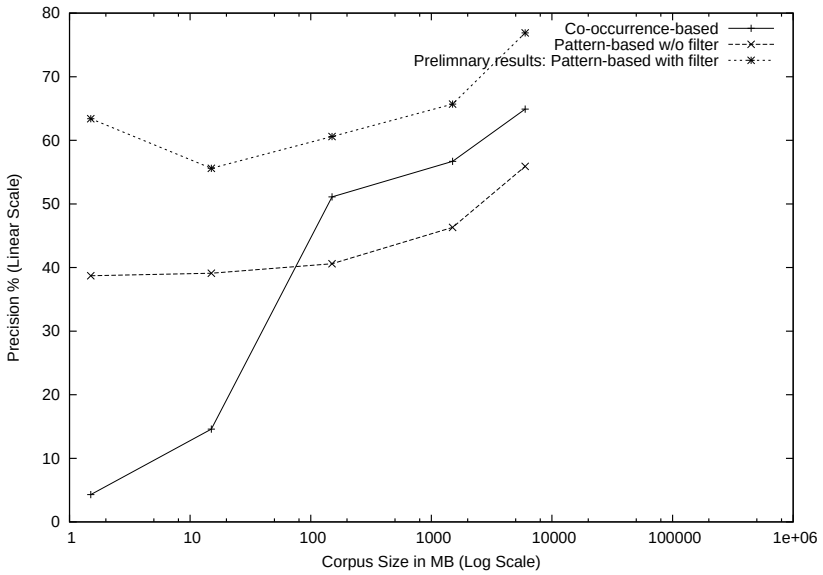

Figure 2: Graph showing the precision of the extracted relations as a function of the corpus size.

proach only four days to complete. It took the pattern-based system 10 days on 15GB corpus.

The results are very encouraging for the linguistically-light pattern based method. The linguistically-rich co-occurrence approach has a problem with respect to scalability. Scaling the entire process to a terabyte holds a lot of promise. We expect to see more relations because proper nouns are potentially an open set and we do learn a lot of proper nouns. The redundancy factor of the knowledge may help to improve precision. We plan to use these extracted relations for knowledge acquisition.

8 Conclusion

In this paper, we explored the various challenges associated with working on terabytes of data. We also made a strong case for working with more data by contrasting two different approaches for extracting *is-a* relations. The shallow pattern based methods with a machine filter has better performance than linguistically rich method. Albeit possible to successfully apply linguistically-light but data-rich approaches to some NLP applications, merely reporting these results often fails to yield insights into the underlying theories of language at play. Our biggest challenge as we venture to the terascale is to use our new found wealth not only to build better systems, but to improve our understanding of language.

References

[1] Banko, Michele and Eric Brill: Mitigating the Paucity of Data Problem. In Proceedings of the *Conference on Human Language Technology*, San Diego, CA. (2001).

[2] Banko, Michele and Eric Brill: Scaling to a Very Very Large Corpora for Natural Language Disambiguation. Proceeding of the *Association for Computational Linguistics*, Toulouse, France. (2001).

[3] Berwick, Robert, Steven Abney, and Carol Tenny, editors: Principle-Based Parsing. *Kluwer Academic Publishers.* (1991).

[4] Brill, Eric: Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part of Speech Tagging. *Computational Linguistics.* (1995).

[5] Brill, E., J. Lin, M. Banko, S. Dumais, and A. Ng: Data-Intensive Question Answering. Proceedings of the *TREC-10 Conference*, pp 183–189. NIST, Gaithersburg, MD. (2001).

[6] Charniak, Eugene: A Maximum-Entropy-Inspired Parser Proceedings of *NAACL*. Seattle, WA. (2000).

[7] Curran, J. and Moens, M.: Scaling context space. In Proceedings of *ACL-02*. pp 231–238, Philadelphia, PA. (2002).

[8] Darroch, J. N., and D. Ratcliff: Generalized iterative scaling for log-linear models. *Annals of Mathematical Statistics*, 43:1470–1480. (1972).

[9] Etzioni, Oren, Michael Cafarella, Doug Downey, Stanley Kok, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates: Web-scale Information Extraction in KnowItAll (Preliminary Results). To appear in the conference on *WWW*. (2004).

[10] Gray, Jim: A Conversation with Jim Gray *ACM Queue* vol. 1, no. 4 – June 2003.

[11] Lin, Dekang: Principar - an Efficient, Broad-Coverage, Principle-Based Parser. In Proceedings of *COLING-94*. pp. 42-48. Kyoto, Japan. (1994).

[12] Pantel, Patrick and Dekang Lin: Discovering Word Senses from Text. In Proceedings of *SIGKDD-02*. pp. 613–619. Edmonton, Canada. (2002).

[13] Pantel, Patrick and Deepak Ravichandran: Automatically Labeling Semantic Classes. In the Proceedings of *NAACL/HLT*, Boston, MA. (2004).

[14] Pantel, Patrick, Deepak Ravichandran, and Eduard Hovy: Towards Terascale Knowledge Acquisition. To appear in the Proceedings of the *COLING* conference, Geneva, Switzerland. (2004).

[15] Och, Franz Josef and Hermann Ney: Discriminative Training and Maximum Entropy Models for Statistical Machine Translation. In Proceedings of *ACL*, pp. 295–302. Philadelphia, PA. (2002).

[16] Siegel, S. and N. J. Castellan Jr.: Nonparametric Statistics for the Behavioral Sciences. *McGraw-Hill*. (1998).

[17] Wang, B., H. Xu, Z. Yang, Y. Liu, X. Cheng, D. Bu, and S. Bai: TREC-10 Experiments at CAS-ICT: Filtering, Web, and QA. Proceedings of the *TREC-10 Conference*, pp 229–241. NIST, Gaithersburg, MD. (2001).