Natural Language Processing

Prof. Pushpak Bhattacharyya
Department of Computer Science and Engineering
Indian Institute of Technology, Bombay

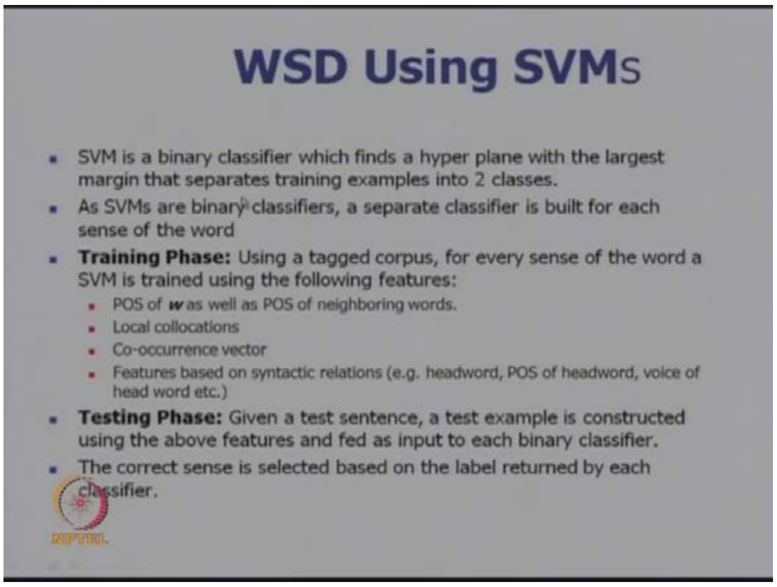
Lecture - 35
Word Sense Disambiguation: Semi-Supervised and Unsupervised Method; Resource-Constrained WSD

We would like to finish the discussion on word sense disambiguation. And touch upon the algorithms which are based on semi supervised and unsupervised learning. In particular, we would like to make some summarising remarks on how supervised semi supervised and unsupervised algorithms relate to each other and how do they compare. Then we will move on to our work on resource constrained word sense disambiguation, which we is a very important line of work for languages which do not have lots of sense mark-up corpora and there is absence of resources for these languages. Indian languages fall in that category and we have had done some pioneering work in the field of semi supervised and unsupervised, word sense disambiguation where the resources are scarce.

So, so far what you have seen is that we have looked at a number of word sense disambiguation algorithms based on the idea of overlap which essentially takes the context words. And then sees the overlap of these works in the sense repository of the wordNet So, one fundamental point to remember is that, the words in the neighbourhood of a target word undergoing disambiguation are the most important clues for word sense disambiguation. So, these words form the feature vectors for supervised machine learning waste algorithm for word sense disambiguation, they form the overlap for knowledge based, overlap based word sense disambiguation algorithms. And also the context clues for unsupervised word sense disambiguation algorithms. So, as is intuitively clear and which probably does not need too much of reiteration that it is the words in the neighbourhood of the target word that have to provide the clues for disambiguation.

So, in the last lecture, we had seen k nearest neighbourhood based word sense disambiguation. We will very briefly look at 2 more algorithms, and then move on to semi supervised and unsupervised algorithms for word sense disambiguation. So, going to the slides, we see that our topic is word sense disambiguation semi supervised and unsupervised methods, and moving on to resource constrained word sense disambiguation.

(Refer Slide Time: 03:40)



WSD Using SVMs

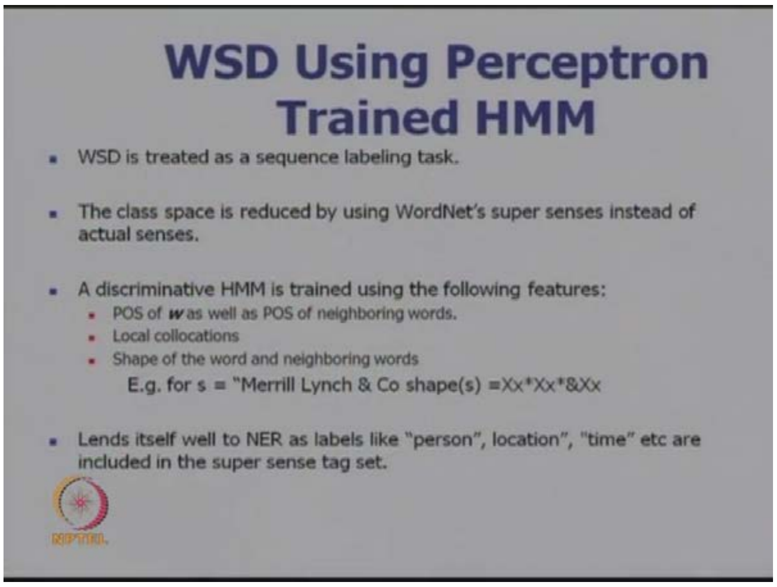
- SVM is a binary classifier which finds a hyper plane with the largest margin that separates training examples into 2 classes.
- As SVMs are binary classifiers, a separate classifier is built for each sense of the word
- **Training Phase:** Using a tagged corpus, for every sense of the word a SVM is trained using the following features:
  - POS of ***w*** as well as POS of neighboring words.
  - Local collocations
  - Co-occurrence vector
  - Features based on syntactic relations (e.g. headword, POS of headword, voice of head word etc.)
- **Testing Phase:** Given a test sentence, a test example is constructed using the above features and fed as input to each binary classifier.
- The correct sense is selected based on the label returned by each classifier.

So, since the words in the neighbourhood of the target word provide the contextual clues they are used to form the feature vector. And once a feature vector is formed then the task is let us say three by fourth done. Because after that it is only the problem of finding a machine learning algorithm, a suitable machine learning algorithm that will place a label on the word in terms of its sense or will place the word in a class the class being a sense for the word. So, if we look at the title of this slide word sense disambiguation using SVMs support vector machines. The support vector machines form a class of powerful classifiers, which find hyper plane with the largest margin that separates training examples into 2 classes. So, this falls in the category of a large class of classifiers which set up hyper planes to classify the positive points from a negative points as SVMs are binary classifiers a separate classifier is built for each sense of the word.

In the training phase, we have to use a tagged corpus for every sense of the word a SVM s is trained using the following feature. Part of speech of w as well as part of speech of neighbouring words local collocations, co-occurrence vectors. Features based on syntactic relations for example, headword, POS of headword, voice of headword and so on. So, as is frenloan in supervised machine learning algorithms, it is the feature vector which is the most important component of the training scenario. The richer, the feature vector; the more effective is the learning algorithm

So, that is why we see that in the training phase there are a number of features of the target word as well as the neighbourhood words, which are considered. In the testing phase given a test sentence a test example is constructed using the above features. And is fed as input to each binary classifier, the correct sense is selected based on the label returned by each classifier. So, this shows what is done in the SVM based classification, SVMs have become very important for text processing, text mining, text information extraction applications. And therefore, it is not a surprise that SVMs are being tried for word sense disambiguation also.

(Refer Slide Time: 07:01)

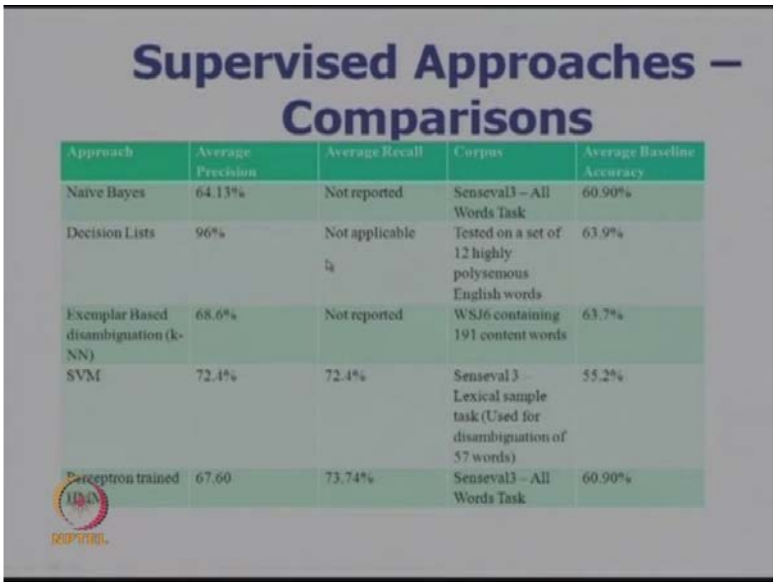


WSD Using Perceptron Trained HMM

- WSD is treated as a sequence labeling task.
- The class space is reduced by using WordNet's super senses instead of actual senses.
- A discriminative HMM is trained using the following features:
  - POS of *w* as well as POS of neighboring words.
  - Local collocations
  - Shape of the word and neighboring words
    E.g. for s = "Merrill Lynch & Co shape(s) =Xx*Xx*&Xx
- Lends itself well to NER as labels like "person", location", "time" etc are included in the super sense tag set.

Now, word sense disambiguation using percept on trained hidden markov model, here the task of word sense disambiguation is looked upon as a sequence labelling task. The class space is reduced by wordNet super senses instead of actual senses. A discriminative hidden markov model is trained using the following features part of speech of word as well as part of speech of neighbouring words local collocations, shape of the word and neighbouring words. And this algorithm is also quite suitable for named entity recognition labels like person location time etcetera. And they make use of a powerful idea called the super sense tag set. So, with this, we will close discussion on supervised machine learning algorithm for word sense disambiguation. And we would like to proceed to semi supervised and unsupervised algorithms.

(Refer Slide Time: 08:11)



Supervised Approaches – Comparisons

Approach	Average Precision	Average Recall	Corpus	Average Baseline Accuracy
Naive Bayes	64.13%	Not reported	Senseval3 – All Words Task	60.90%
Decision Lists	96%	Not applicable	Tested on a set of 12 highly polysemous English words	63.9%
Exemplar Based disambiguation (k-NN)	68.6%	Not reported	WSJ6 containing 191 content words	63.7%
SVM	72.4%	72.4%	Senseval 3 – Lexical sample task (Used for disambiguation of 57 words)	55.2%
Perceptron trained HMM	67.60	73.74%	Senseval3 – All Words Task	60.90%

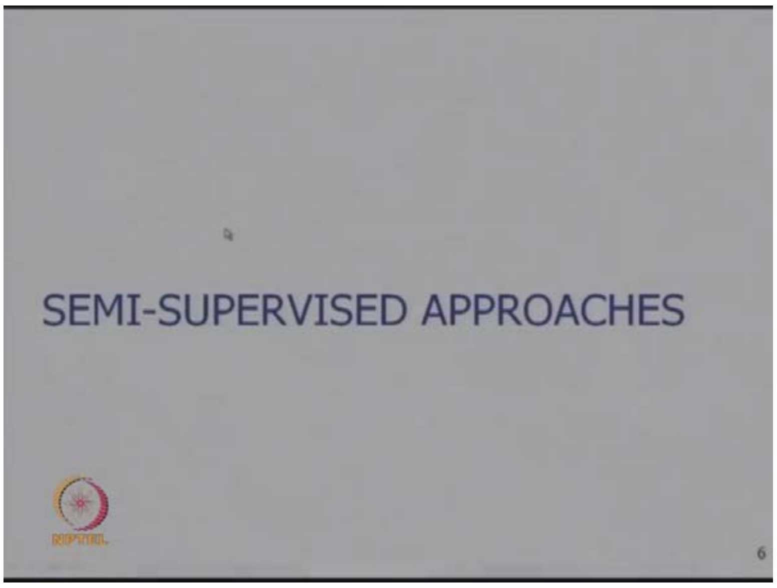
But before that a table on remarks on comparison of supervised approached to word sense disambiguation. So, if the approach is Naïve Bayes then the average precision is seen to be about 64 percent, average recall is not reported in the literature. The corpus used is senseval3 all words task and the average baseline accuracy for this task is about 61 percent. So, the Naïve Bayes algorithm improves on the performance of baseline by about 3 percent. If the approach is decision lists based then we find the average precision is very high 96 percent. Average recall; this question is not really applicable here the corpus was used such that there were 12 highly polysemous english words and the testing was done on that. And the baseline accuracy is 64 percent and therefore, the average precision is really very high compared to the baseline accuracy.

Exemplar based disambiguation which is k nearest neighbour precision is 69 percent or so recall is not reported. And the corpus is wall street journal corpus using 198 191 content words, the baseline accuracy is reported to be 64 percent. So, there is about 5 percent improvement in accuracy when one uses this algorithm. For SVM based approach the accuracy is 72 percent, sensible corpus is used and 57 target towards or seen. And we find the average based on accuracy is 55 percent. Therefore, the improvement is in accuracy is about 27 percent, no, 17 percent

Then perceptron trained HMM gives an accuracy of about 67 percent, baseline is 61 percent, corpus use is senseval3 is all words disambiguation task. And again the accuracy improvement is about 7 to 8 percent. So, what do we summarise from these observations, we find that supervised learning algorithms, go to achieve the accuracy of about 65 percent or there about. And therefore, this is a good powerful approach for word sense disambiguation, any performance above 65 percent is good in word sense disambiguation.

However, as has been said the constrained of using supervised algorithm for word sense disambiguation is that we need large about of training corpus. And that is time consuming labour intensive, money intensive and it is not an easy matter. So, can we do something about these very variable resource called sense smart corpus, can we. For example, not have a very large amount of training corpus but start with a seed amount can we do completely away with training corpus. So, these possibilities need to be examined and that is the next part of the discussion.

(Refer Slide Time: 12:10)

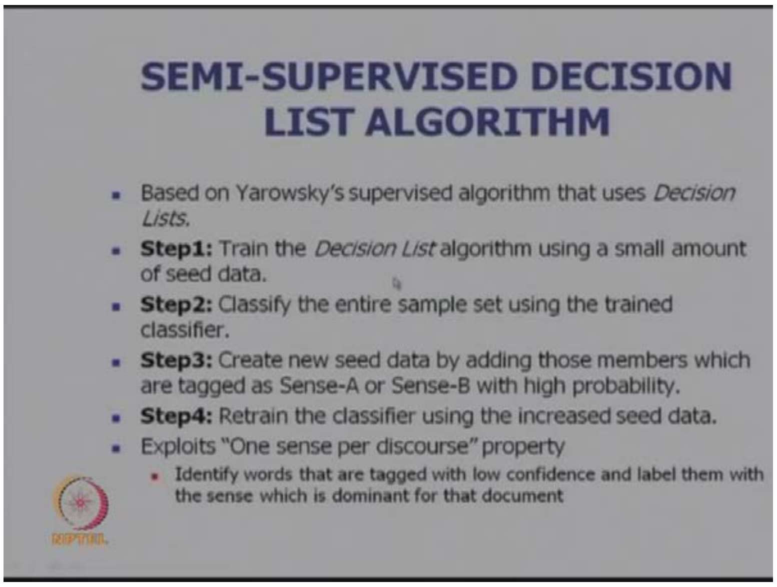


SEMI-SUPERVISED APPROACHES

6

So, semi supervised approaches.

(Refer Slide Time: 12:13)



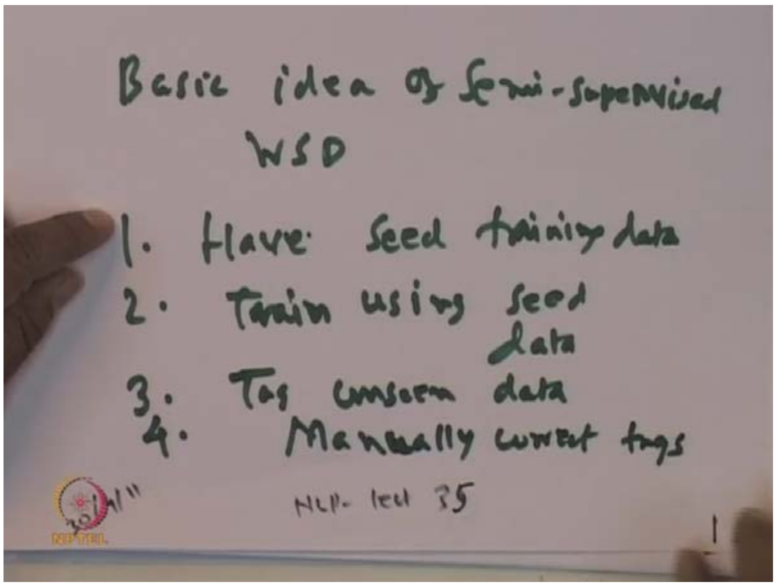
SEMI-SUPERVISED DECISION LIST ALGORITHM

- Based on Yarowsky's supervised algorithm that uses *Decision Lists*.
- **Step1:** Train the *Decision List* algorithm using a small amount of seed data.
- **Step2:** Classify the entire sample set using the trained classifier.
- **Step3:** Create new seed data by adding those members which are tagged as Sense-A or Sense-B with high probability.
- **Step4:** Retrain the classifier using the increased seed data.
- Exploits "One sense per discourse" property
  - **Identify words that are tagged with low confidence and label them with the sense which is dominant for that document**

The first algorithm is semi supervised decision lists algorithm, it is based on Yarowsky's supervised algorithm that uses the decision lists. So, we need to train the decision lists algorithm using a small amount of seed data. So, in all semi supervised learning algorithms, the there is one uniform idea. The idea is that we use a small amount of training corpus; this training corpus is used to train a machine for the task involved. After the machine is trained it is used to tag the unseen corpus the unseen corpus is tagged.

But there surely is some amount of error, because the training was done on very small amount of data. And it was not possible to look at every region of the concept space. So, semi supervised learning algorithm typically entails a phase of tagging followed by manual correction. Once manual correction is done, the machine is retrained with larger amount of training data. And this procedure continues tagging manual correction retraining this cycle continues until we achieve the desired level of accuracy. So, this we can summarise by writing on a piece of paper.

(Refer Slide Time: 14:04)



Basic idea of Semi-supervised WSD

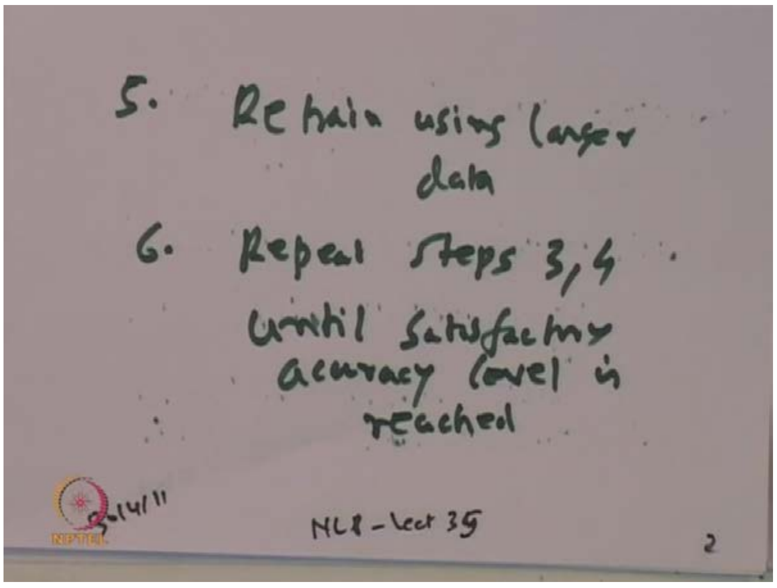
1. Have seed training data
2. Train using seed data
3. Tag unseen data
4. Manually correct tags

NLP- lect 35

1

Basic idea of semi supervised WSD is that have seed training data, tag no first train using seed data then tag unseen data manually correct tags.

(Refer Slide Time: 15:08)



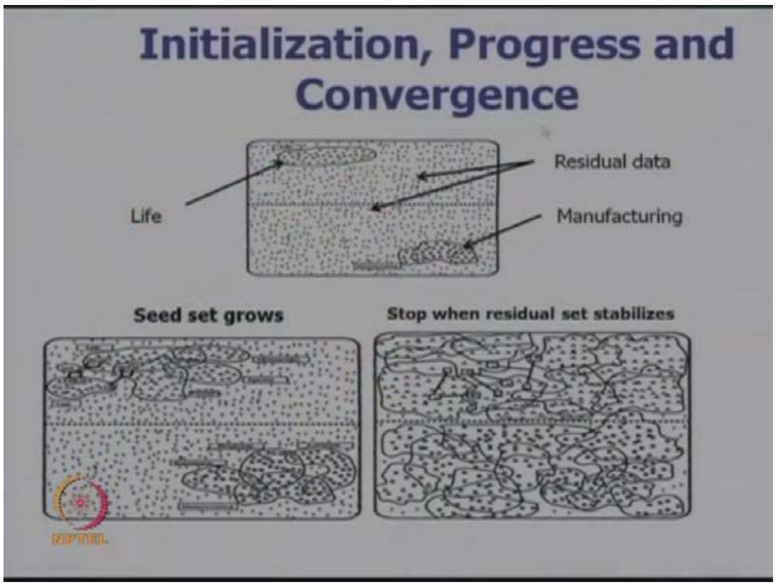
5. Retrain using larger data
6. Repeat steps 3, 4 until satisfactory accuracy level is reached

NLP - lect 35

2

Step 5 retrain using larger data repeat steps 3, 4 until satisfactory accuracy level is reached. This is the main idea of semi supervised learning algorithm in any domain and word sense disambiguation also follows the same path, now a diagram on the slide.

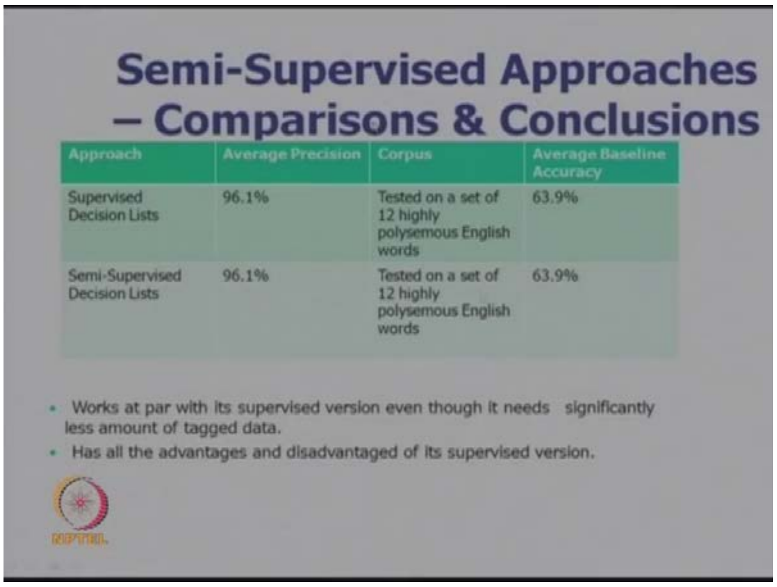
(Refer Slide Time: 16:06)



Initialization, Progress and Convergence

Shows this picture initialisation progress and convergence, so this has similarity with for example, how vegetables grow from seeds? So, there is a small amount of seed data and that is used to populate terrain using the seed data and we obtain larger vegetation the seed set gradually grows. And we stop when the residual set stabilises or when the learning algorithm correspondingly converges to acceptable level of accuracy.

(Refer Slide Time: 16:55)



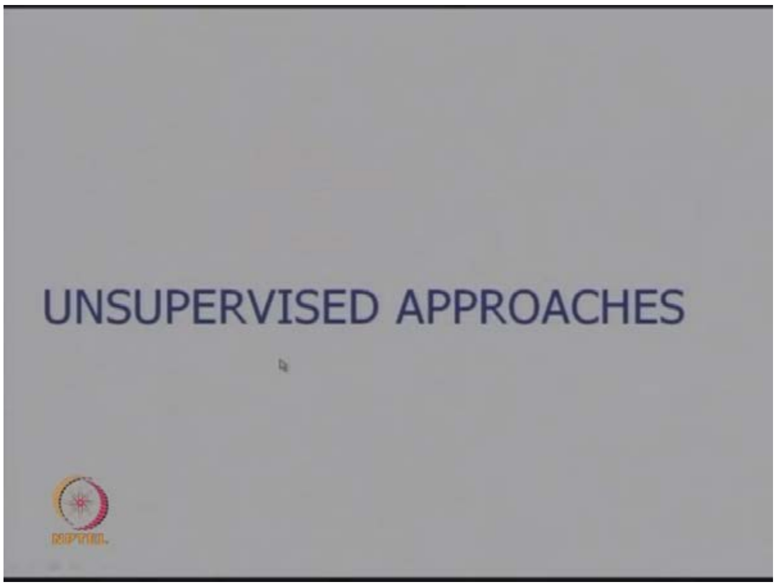
Semi-Supervised Approaches – Comparisons & Conclusions

Approach	Average Precision	Corpus	Average Baseline Accuracy
Supervised Decision Lists	96.1%	Tested on a set of 12 highly polysemous English words	63.9%
Semi-Supervised Decision Lists	96.1%	Tested on a set of 12 highly polysemous English words	63.9%

- Works at par with its supervised version even though it needs significantly less amount of tagged data.
- Has all the advantages and disadvantaged of its supervised version.

Some remarks on semi supervised approaches are the following. Semi supervised decision lists have an average precision of 96 percent tested on a set of 12 highly polysemous English words. The baseline accuracy is about 64 percent, semi supervised decision lists give that kind of accuracy 96.1 percent and tested on the same data average baseline accuracy of course, remains the same. And what we find that semi supervised decision list algorithm works at par with the supervised version and it needs significantly less amount of tagged data.

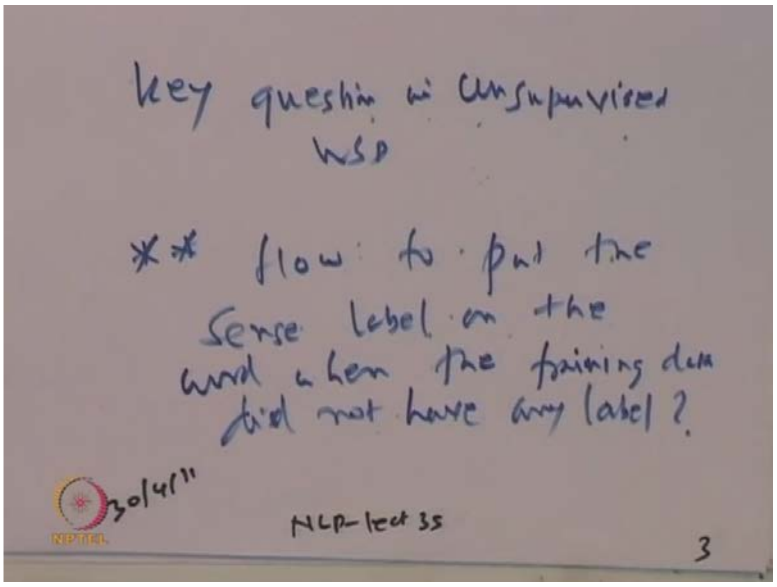
(Refer Slide Time: 16:55)



UNSUPERVISED APPROACHES

So, proceeding further, we go to the topic of unsupervised approaches to word sense disambiguation. The attraction of unsupervised approaches is that there is no dearth of electronic corpora in any language. However creation of sense smart corpus is expensive as has been already said. So, there has been a large body of work exploring whether or not it is possible to do word sense disambiguation in an unsupervised setting. The key question there is the following, we will write it down.

(Refer Slide Time: 18:26)



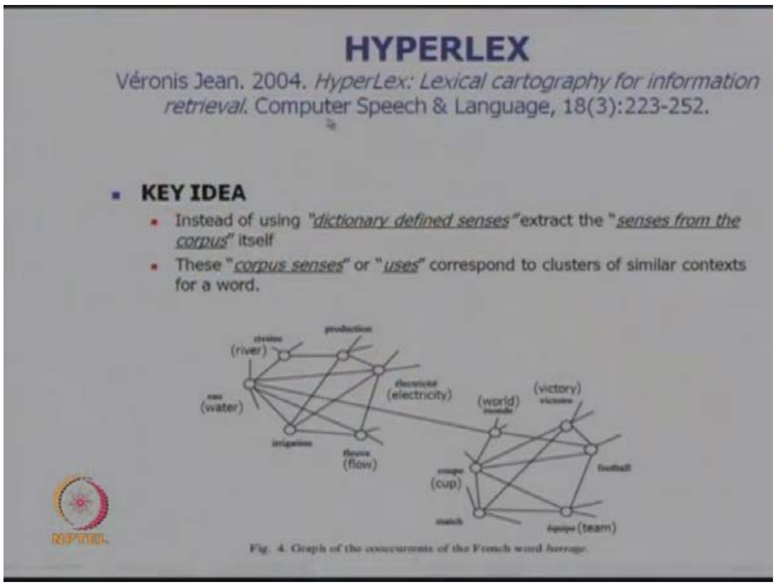
key question in unsupervised WSD

** How to put the sense label on the word when the training data did not have any label?

30/4/11
NLP-lect 35
3

Key question in unsupervised WSD, we will put a star on this. How to put the sense label on the word when the training data did not have any label? It is a key question. How to put the sense label on the word when the training data did not have any label? So, that is a key question in any unsupervised setting when we have not used any label in the training phase, how do we suddenly invent labels to place on the words. So, this problem is solved by having a lexicon of labels. They are not used to mark the training to mark the training data they are used to produce the labels once we have determined the sense of the word. So, there is this question of going from the decision of the sense on the word to the label on the word, this bridge needs to be closed; this gap needs to be closed.

(Refer Slide Time: 20:24)



HYPERLEX

Véronis Jean. 2004. *HyperLex: Lexical cartography for information retrieval.* Computer Speech & Language, 18(3):223-252.

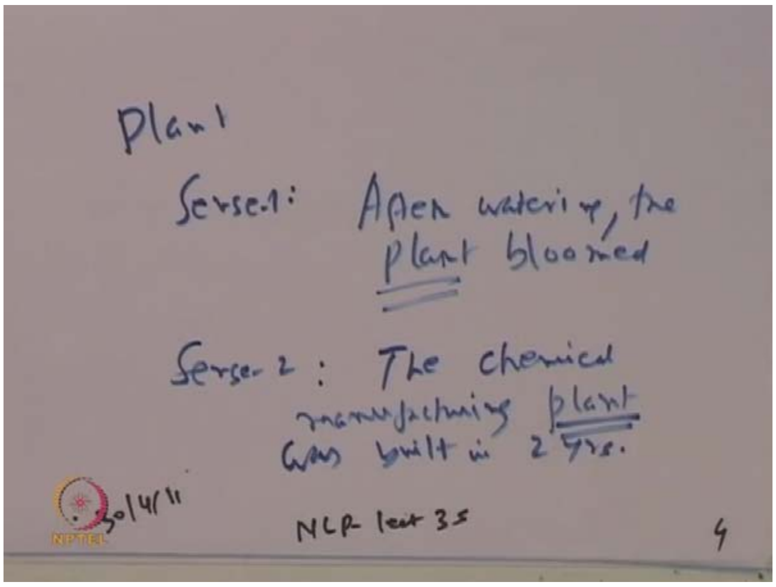
- **KEY IDEA**
  - Instead of using "*dictionary defined senses*" extract the "*senses from the corpus*" itself
  - These "*corpus senses*" or "*uses*" correspond to clusters of similar contexts for a word.

Fig. 4. Graph of the cooccurrents of the French word barrage.

So, for the unsupervised approach, we discussed a well-known work called the hyperlex algorithm for unsupervised WSD, this was published in 2004 in the journal computer speech and language, the author is Jean Veronis. So, the key idea here is described instead of using dictionary defined senses extract the senses from the corpus itself. These corpus senses or uses correspond to clusters of similar contexts for a word. So, this idea, we have met before; we have met before in the supervised setting.

The idea was used in decision lists based word sense disambiguation. We had taken this example of plant where plant has the meaning of manufacturing or it has the meaning of vegetation life flora. So, plant has one meaning of being a small tree or being a manufacturing plant. Now, in this case the decision list was formed using the collocations the words in the context. However, there a label was used after watering the plant, it slowly bloomed after watering the plant; it slowly bloomed so let me write it down.

(Refer Slide Time: 22:08)


Plant

Sense 1: After watering, the plant bloomed

Sense 2: The chemical manufacturing plant was built in 2 yrs.

30/4/11

NLP lect 35

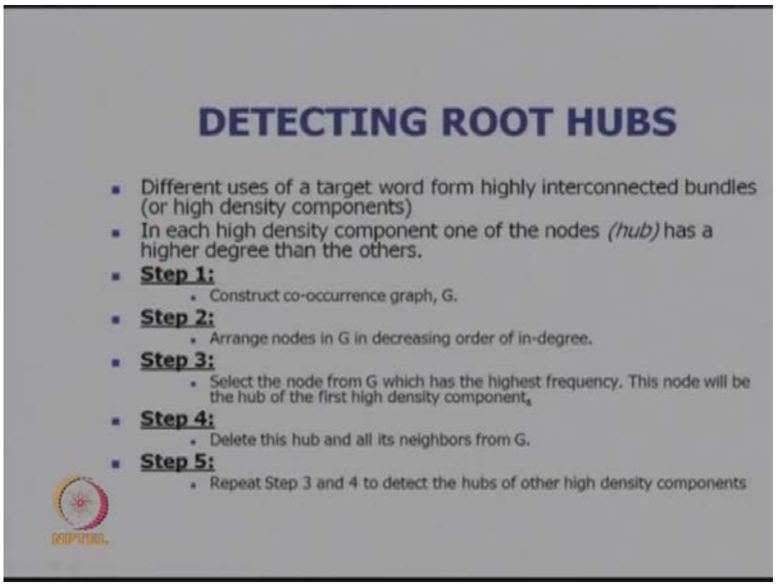
4

The word is plant sense 1; after watering the plant bloomed. Sense 2; the chemical manufacturing plant was built in 2 years. So, the, what we disambiguated is plant and the very important clues for sense 1 are water and bloom, very important clues for plant here is manufacturing built chemical and so on. Now, in unsupervised learning algorithm, what we will do is that if at all we find in the test sentence these words coming in the context of plant water bloom etcetera in the test sentence then we latch on to this sense, sense 1. And opposite will be the case, if we find chemical manufacture built such words in the neighbourhood of the word plant. This is a simple idea and different algorithms, try to capture these neighbouring words as contextual clues.

So, the same thing is expressed here in the slide key idea is that instead of using dictionary defined senses, extract the senses from the corpus itself which is nothing but the context words. These corpus senses or uses correspond to clusters of similar contexts for a word the (( )) discusses the idea with a French example. The example here is water; the French word is eau. So, first of all what happens is that, we create a graph with collocating words. So, this shows a connecting component of the graph where collocated words like river, water, production, irrigation, flow, electricity, etcetera, come together.

There is another connected component in the graph from the contextual words where we find the world cup match team football victory etcetera coming together. So, you can see that the world cup which has many, many different senses. Here cup means a trophy, these senses come together or the sense comes in association with senses of words like victory, team, football, world and so on. And the water sense of the word EAU comes with the words river, irrigation, production, electricity, flow which together form a coherent sense graph.

(Refer Slide Time: 25:59)

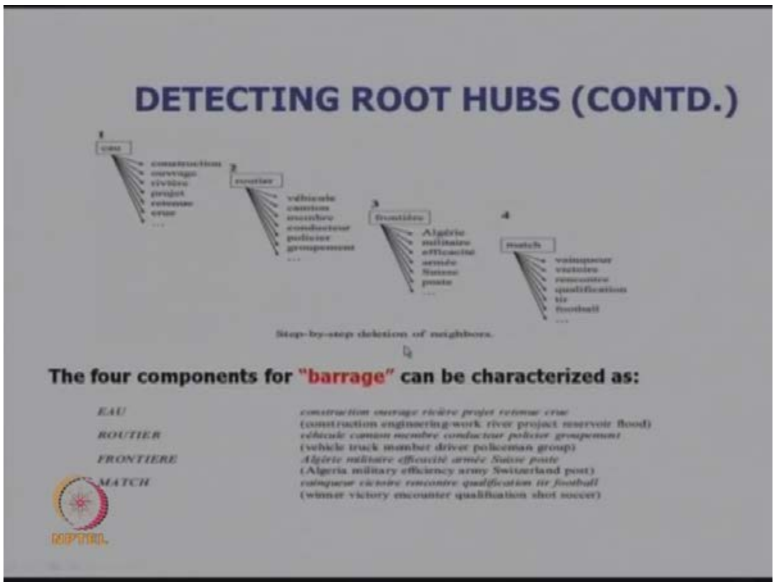


DETECTING ROOT HUBS

- Different uses of a target word form highly interconnected bundles (or high density components)
- In each high density component one of the nodes *(hub)* has a higher degree than the others.
- **Step 1:**
  - Construct co-occurrence graph, G.
- **Step 2:**
  - Arrange nodes in G in decreasing order of in-degree.
- **Step 3:**
  - Select the node from G which has the highest frequency. This node will be the hub of the first high density component.
- **Step 4:**
  - Delete this hub and all its neighbors from G.
- **Step 5:**
  - Repeat Step 3 and 4 to detect the hubs of other high density components

Now, the target words give rise to what is called the root hubs. Different uses of a target word form highly interconnected bundles or high density components. And in each high density component one of the nodes has a higher degree than the others. So, first we construct the co-occurrence graph that means we take the words which are contextually related, related meaning wise also. Arrange the nodes in G in decreasing order of in-degree. Select the node from G which has the highest frequency. This node will be the hub of the first high density component. Delete this hub and all its neighbours from G. Repeat step 3 and 4 to detect the hubs of other high density components.

(Refer Slide Time: 26:54)



DETECTING ROOT HUBS (CONTD.)

1
eau
construction
ouvrage
rivière
projet
retenue
crue
...
2
routier
véhicule
camion
membre
conducteur
policier
groupement
...
3
frontière
Algérie
militaire
efficacité
armée
Suisse
poste
...
4
match
vainqueur
victoire
rencontre
qualification
tir
football
...

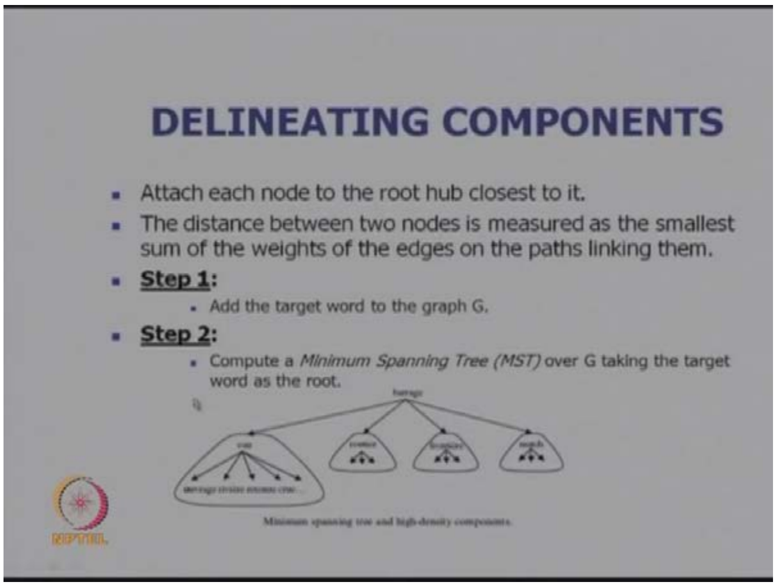
Step-by-step deletion of neighbors.

The four components for "barrage" can be characterized as:

EAU	*construction ouvrage rivière projet retenue crue* (construction engineering-work river project reservoir flood)
ROUTIER	*véhicule camion membre conducteur policier groupement* (vehicle truck member driver policeman group)
FRONTIERE	*Algérie militaire efficacité armée Suisse poste* (Algeria military efficiency army Switzerland post)
MATCH	*vainqueur victoire rencontre qualification tir football* (winner victory encounter qualification shot soccer)

So, here is an example of the word barrage which is getting disambiguated the word barrage has 4 components in the sense of sense association graphs. So, one component is with the sense of water where we find the words construction, engineering, work, river, project, reservoir, flood coming together. The other component has the sense of vehicle and associated senses vehicle, truck, member, driver, policeman group they come together. Then there is association on the frontier sense where we have Algeria, military efficiency, army, Switzerland post these are very strongly associated words they come together to form a sense component. Then match here the correlated words are winner, victory, and counter, qualification, shot, soccer, etcetera. So, we find that the word barrage can participate in one of these four connected components as sense sub-graphs.

(Refer Slide Time: 28:26)


DELINEATING COMPONENTS

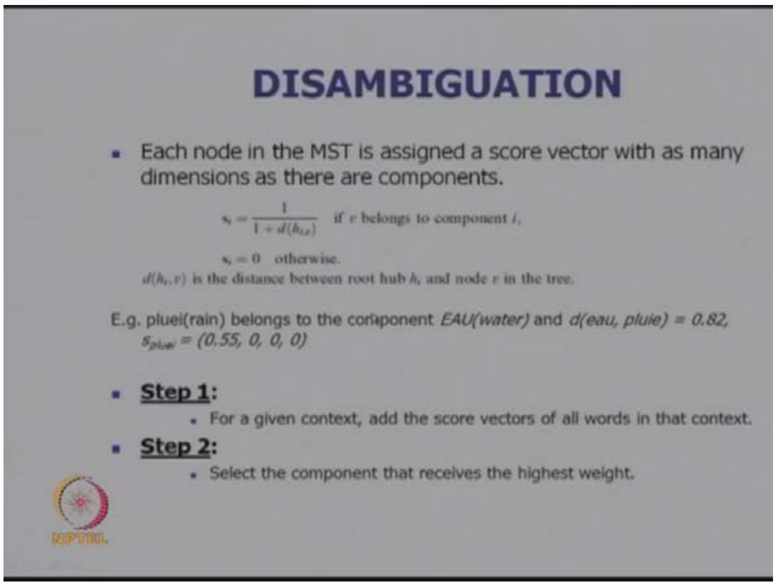
- Attach each node to the root hub closest to it.
- The distance between two nodes is measured as the smallest sum of the weights of the edges on the paths linking them.
- **Step 1**:
  - Add the target word to the graph G.
- **Step 2**:
  - Compute a *Minimum Spanning Tree (MST)* over G taking the target word as the root.

barrage
eau

Minimum spanning tree and high-density components.

Then there is a stage of delineating the components attached each node to the root hub closest to it. The distance between 2 nodes is measured as the smallest sum of the weights of the edges on the path linking them the concept of weights will be explained in the discussion coming after 2 or 3 slides. So, the delineation involves first step at the target word to the graph G. And step 2 is compute a minimum spanning tree over G taking the target word as a root. So, spanning tree, it would be remembered is a part of the graph which covers all the vertices, but may not cover all the edges. So, this is a part of the graph which is cycle free now what happens is that.

(Refer Slide Time: 29:27)


DISAMBIGUATION

- Each node in the MST is assigned a score vector with as many dimensions as there are components.

$$s_i = \frac{1}{1 + d(h_{i,v})} \quad \text{if } v \text{ belongs to component } i,$$

$$s_i = 0 \quad \text{otherwise.}$$

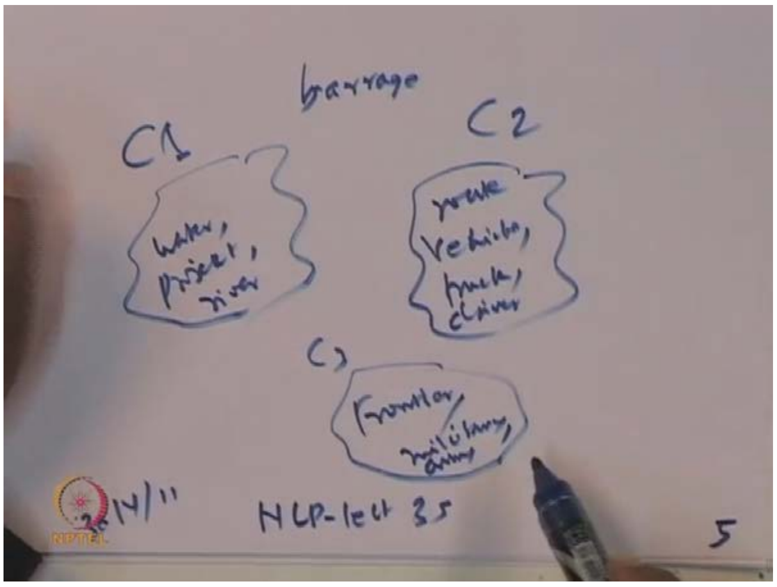
$d(h_i, v)$ is the distance between root hub h_i and node v in the tree.

E.g. pluei(rain) belongs to the component *EAU(water)* and *d(eau, pluie) = 0.82*, *s_{pluei} = (0.55, 0, 0, 0)*

- **Step 1**:
  - For a given context, add the score vectors of all words in that context.
- **Step 2**:
  - Select the component that receives the highest weight.

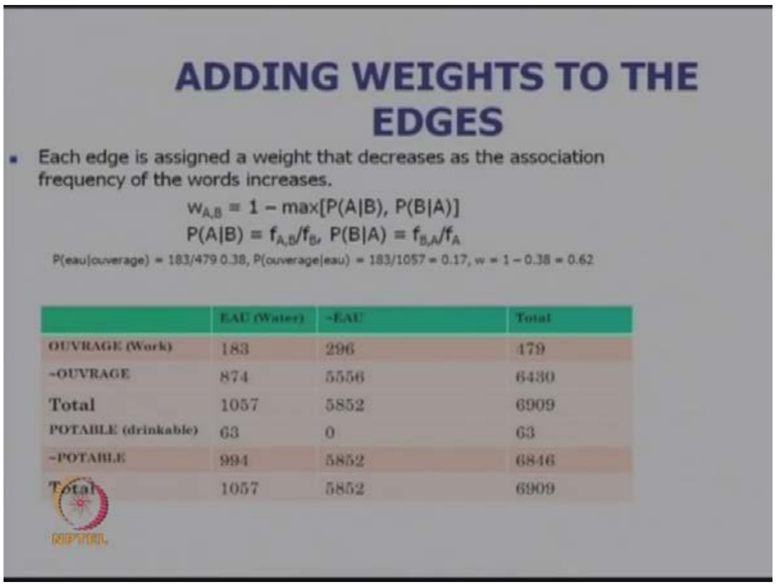
After delineation, we discuss disambiguation each node in the minimum spanning tree is assigned a score vector with as many dimensions as there are components so barrage for example, had four components there are these formulae which are empirical in nature And we will not go into the details of how these are obtained, but some amount of intuition can be given d h i, v is the distance between root hub h i and the node v in the tree. So, the sense of the word v is S i which is equal to one by one plus d h i comma v if v belongs to the component i otherwise it is 0. So, it is easy to see what is going here we find the distance of the word from the cluster. So, we explain it diagrammatically. So, what is happening here is that if I take the word barrage.

(Refer Slide Time: 30:42)



So, there is a component with water, project, river, and so on. Let us call this component C 1. Another component with vehicle, truck, driver, and so on, this is component C 2. Another component C 3 with the sense of frontier and it has military then army and so on. Now, when this word barrage is seen, we see from the context words of barrage in the test sentence which component it is close to. Suppose the context words of barrage in the sentences such that the distance of the neighbourhood words and of barrage from these from among these components is a list for C 2. Then we know that we have the vehicle, truck, driver, etcetera as the strong association of barrage and root sense is the winning sense for the word barrage. So, this is really a cluster based algorithm and inter-cluster or distance from the cluster is the guiding factor for the algorithm. The minimum spanning tree essentially facilitates finding the distance of the word in that context from a particular component.

(Refer Slide Time: 33:06)



ADDING WEIGHTS TO THE EDGES

- Each edge is assigned a weight that decreases as the association frequency of the words increases.

$$w_{A,B} = 1 - \max[P(A|B), P(B|A)]$$

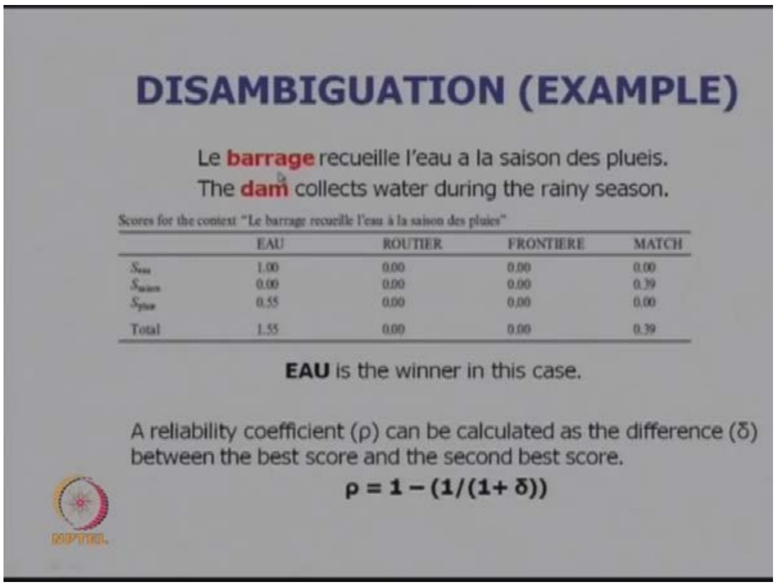
$$P(A|B) = f_{A,B}/f_B,\ P(B|A) = f_{B,A}/f_A$$

P(eau|ouverage) = 183/479 0.38, P(ouverage|eau) = 183/1057 = 0.17, w = 1 – 0.38 = 0.62

	EAU (Water)	~EAU	Total
OUVRAGE (Work)	183	296	479
~OUVRAGE	874	5556	6430
Total	1057	5852	6909
POTABLE (drinkable)	63	0	63
~POTABLE	994	5852	6846
Total	1057	5852	6909

So, now we discuss the weight value, weight value is 1 minus maximum of P A given B comma P B given A which is a probability value. The probability of a given B is computed as frequency of A and B together divided by frequency of B. And probability of B given A is frequency of B and A together divided by frequency of A. So, here the probability of EAU given overage is 183 by 479 which comes out to be equal to 0.38 probability of average given EAU is 183 by 1057 which comes out to 0.17. So, weightage is equal to 1 minus 0.38 which is 0.62. And this table shows the values for other components for the senses in discussion. And from this, it is possible to identify the distance from the sense cluster in which we are interested.

(Refer Slide Time: 34:28)



DISAMBIGUATION (EXAMPLE)

Le **barrage** recueille l'eau a la saison des plueis.
The **dam** collects water during the rainy season.

Scores for the context "Le barrage recueille l'eau à la saison des pluies"

	EAU	ROUTIER	FRONTIERE	MATCH
S_{eau}	1.00	0.00	0.00	0.00
S_{saison}	0.00	0.00	0.00	0.39
S_{pluie}	0.55	0.00	0.00	0.00
Total	1.55	0.00	0.00	0.39

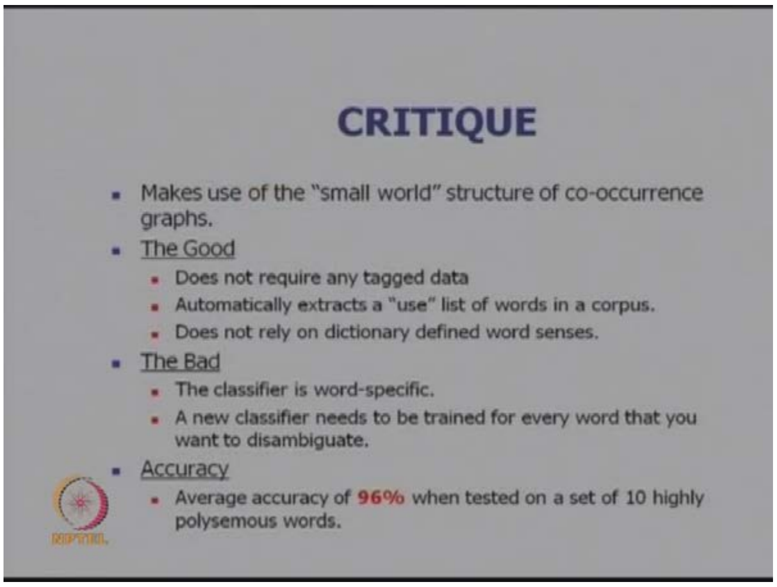
EAU is the winner in this case.

A reliability coefficient (ρ) can be calculated as the difference (δ) between the best score and the second best score.

$$\rho = 1 - (1/(1+\delta))$$

So, here is an example for the word barrage is the French sentence is given here and the English sentence is the dam collects water during the rainy season. So, here we find that water is the winner in this case. So, the component which contains water is the winning sense and that determines the sense of the word barrage.

(Refer Slide Time: 34:58)

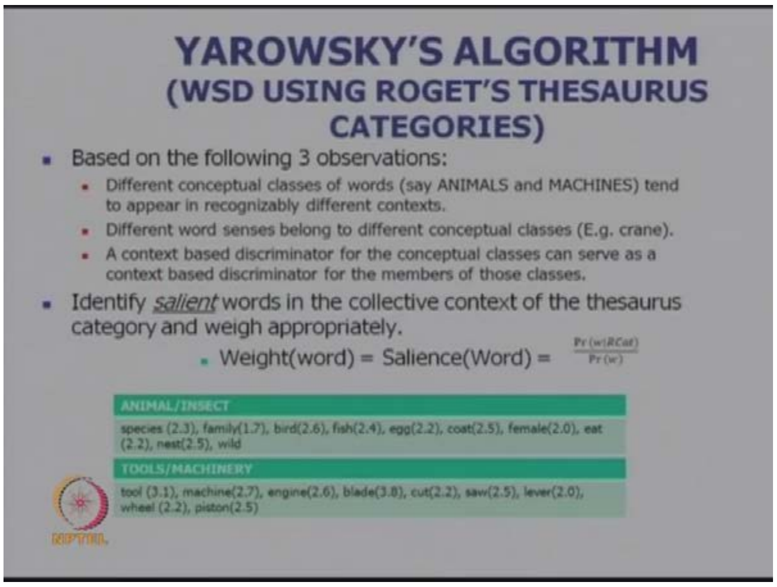


CRITIQUE

- Makes use of the "small world" structure of co-occurrence graphs.
- The Good
  - Does not require any tagged data
  - Automatically extracts a "use" list of words in a corpus.
  - Does not rely on dictionary defined word senses.
- The Bad
  - The classifier is word-specific.
  - A new classifier needs to be trained for every word that you want to disambiguate.
- Accuracy
  - Average accuracy of **96%** when tested on a set of 10 highly polysemous words.

So, some of the critics of this algorithm is that it makes use of the small world structure of co-occurrence graphs. The good part is that it does not require any tagged data, automatically extracts a use list of words in a corpus does not rely on dictionary defined word senses because there is no training corpus. The bad is the classifiers is word specific a new classifier needs to be trained for every word that we want to disambiguate the average accuracy is 96 percent when tested on a set of ten highly polysemous words.

(Refer Slide Time: 35:37)



YAROWSKY'S ALGORITHM (WSD USING ROGET'S THESAURUS CATEGORIES)

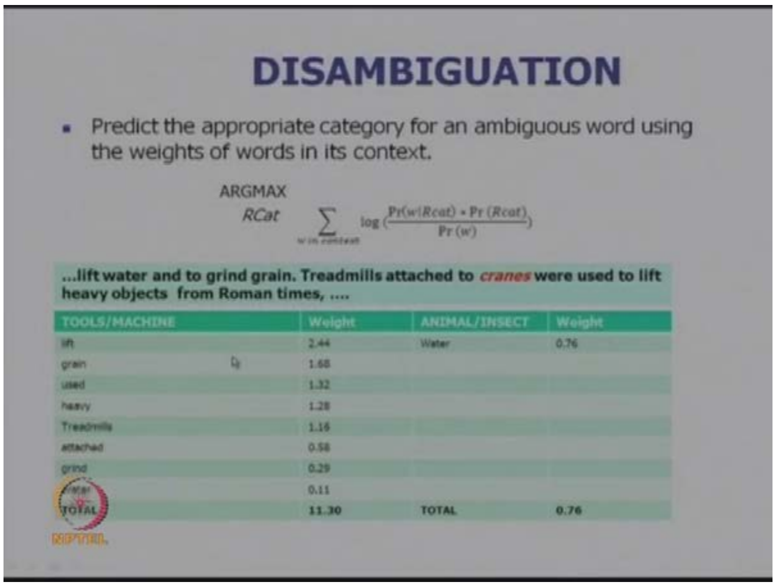
- Based on the following 3 observations:
  - Different conceptual classes of words (say ANIMALS and MACHINES) tend to appear in recognizably different contexts.
  - Different word senses belong to different conceptual classes (E.g. crane).
  - A context based discriminator for the conceptual classes can serve as a context based discriminator for the members of those classes.
- Identify *salient* words in the collective context of the thesaurus category and weigh appropriately.
  - Weight(word) = Salience(Word) = $\frac{Pr(w|RCat)}{Pr(w)}$

ANIMAL/INSECT
species (2.3), family(1.7), bird(2.6), fish(2.4), egg(2.2), coat(2.5), female(2.0), eat (2.2), nest(2.5), wild
TOOLS/MACHINERY
tool (3.1), machine(2.7), engine(2.6), blade(3.8), cut(2.2), saw(2.5), lever(2.0), wheel (2.2), piston(2.5)

There is another famous algorithm on supervised WSD. This makes use of the Roget's thesaurus categories. It is based on the following 3 observations different conceptual classes of words animals and machines tend to appear in recognizably different contexts. Different word senses belong to different conceptual classes for example, crane which has the sense of machine and bird. A context based discriminator for the conceptual classes can serve as a context free discriminator for the members of those classes.

So, we first identify the salient words in the collective context of the thesaurus category and give the weight appropriately. Weight of a word is salience of the word which is the probability of the word given the category divided by the probability of the word. So, here are some examples words like species family bird fish egg coat etcetera they are shown. And we give the salience values in the bracket which are based on the probabilities which in turn are based on frequencies. Similarly there are words in the tools and machinery domain along with their salience values.

(Refer Slide Time: 37:03)



DISAMBIGUATION

- Predict the appropriate category for an ambiguous word using the weights of words in its context.

$$\underset{RCat}{\text{ARGMAX}} \sum_{w \text{ in context}} \log\left(\frac{\Pr(w|Rcat) * \Pr(Rcat)}{\Pr(w)}\right)$$

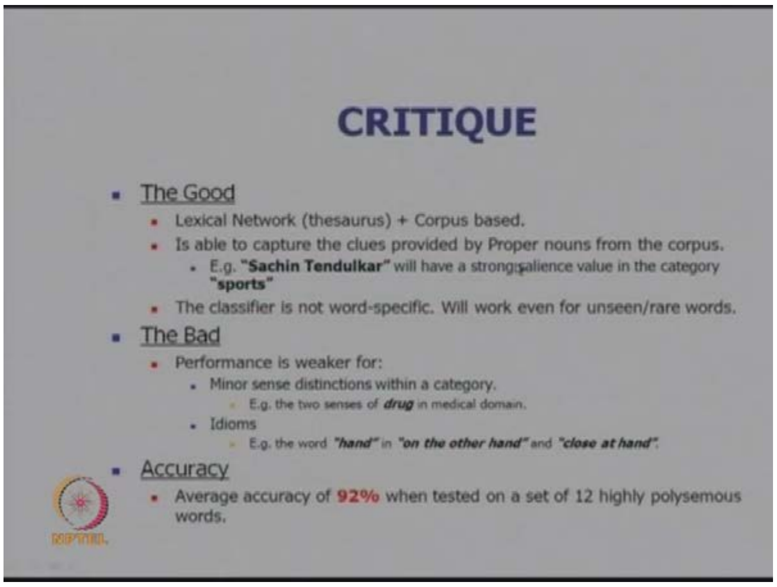
...lift water and to grind grain. Treadmills attached to *cranes* were used to lift heavy objects from Roman times, ...

TOOLS/MACHINE	Weight	ANIMAL/INSECT	Weight
lift	2.44	Water	0.76
grain	1.68		
used	1.32		
heavy	1.28		
Treadmills	1.16		
attached	0.58		
grind	0.29		
water	0.11		
TOTAL	**11.30**	**TOTAL**	**0.76**

So, here how does the disambiguation algorithm work? Predict the appropriate category for an ambiguous word using the weights of the words in the context. So, we have to obtain the category and ARGMAX is based on the category. The base theorem is applied therefore, we have pro probability probability of R cat into the likelihood probability, probability of word given the Roget's category. Here is an example lift water and to grind grain treadmill attached to cranes were used to lift heavy objects from roman times.

So, here the word to be disambiguated is crane, and we find that in the context there are very strong words like water, grind, grain, treadmill, lift heavy objects and so on. So, we take the words from the tools and machine domain. The word lift has a weightage of 2.44, grain has a weightage of 1.68, used has a weightage of 1.32, heavy as a weightage of 1.28 and so on. And crane can also belong to the animal insect domain where we have the contextual clue words like water. So, water for example, is drunk by animals and insects and the weightage for this word is 0.76. So, the weightage combined weightage of clue words coming from machine domain is 11.30, combined weightage of clue words coming from the animal insect domain is 0.76. So, clearly this is the winning value and the domain automatically shifts to the machine domain for the word crane.

(Refer Slide Time: 39:06)



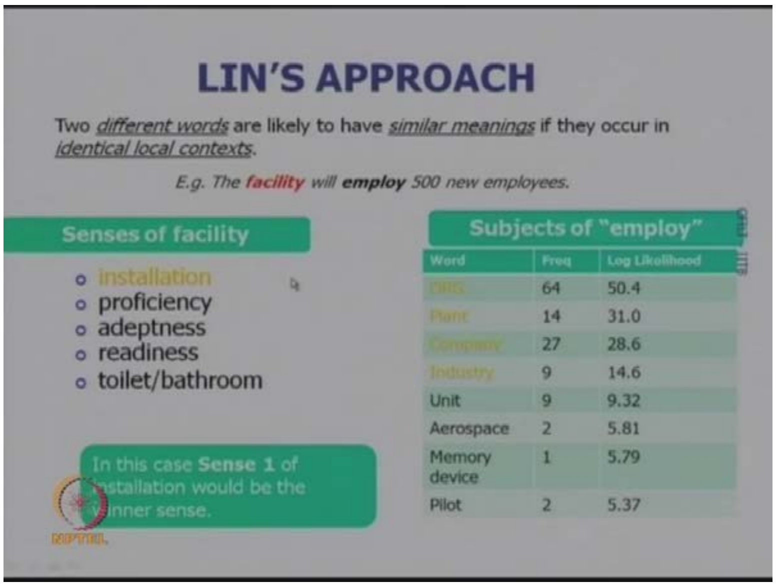
CRITIQUE

- The Good
  - Lexical Network (thesaurus) + Corpus based.
  - Is able to capture the clues provided by Proper nouns from the corpus.
    - E.g. **"Sachin Tendulkar"** will have a strong salience value in the category **"sports"**
  - The classifier is not word-specific. Will work even for unseen/rare words.
- The Bad
  - Performance is weaker for:
    - Minor sense distinctions within a category.
      - E.g. the two senses of ***drug*** in medical domain.
    - Idioms
      - E.g. the word ***"hand"*** in ***"on the other hand"*** and ***"close at hand"***.
- Accuracy
  - Average accuracy of **92%** when tested on a set of 12 highly polysemous words.

The remarks on this particular algorithm which was quite famous at some point of time is that there are good points about this algorithm lexical network that is thesaurus plus corpus is used. So, it is an unsupervised learning algorithm, but making use of a lexical resource is able to capture the clues provided by proper nouns from the corpus. For example, Sachin Tendulkar will have a strong salient value in the category sports. The classifier is not word specific will work even for unseen or rare words.

The bad aspects of this algorithm are the performance is weaker for minor sense distinctions within a category for example, 2 senses of drug in medical domain. So, if the if the polysemous such that there is very little sense distinction between the 2 senses these are shades of senses. Then this algorithm does not perform very well, it does not perform very well for idioms. For example, the word hand in on the other hand or close at hand the algorithm will not give the correct sense. The average accuracy is 92 percent when tested on a set of 12 highly polysemous word.

(Refer Slide Time: 40:21)



LIN'S APPROACH

Two *different words* are likely to have *similar meanings* if they occur in *identical local contexts*.

E.g. The ***facility*** *will* ***employ*** *500 new employees.*

Senses of facility

- installation
- proficiency
- adeptness
- readiness
- toilet/bathroom

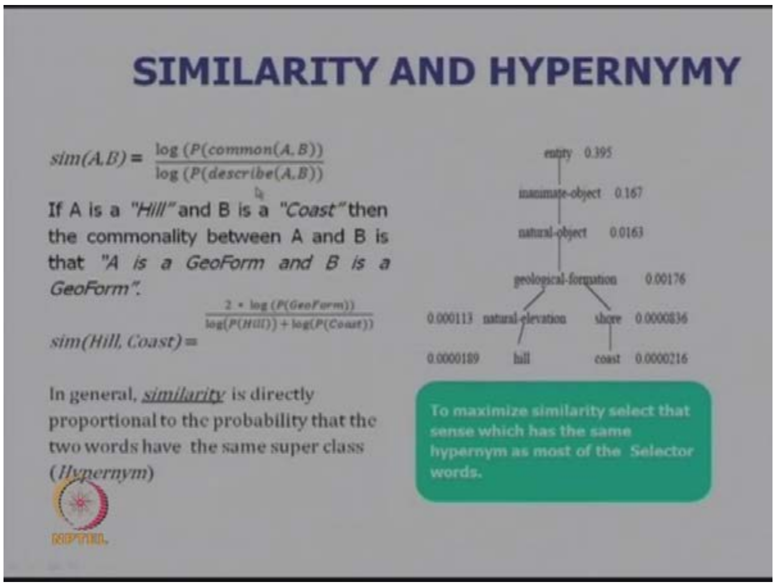
In this case **Sense 1** of installation would be the winner sense.

Subjects of "employ"

Word	Freq	Log Likelihood
ORG	64	50.4
Plant	14	31.0
Company	27	28.6
Industry	9	14.6
Unit	9	9.32
Aerospace	2	5.81
Memory device	1	5.79
Pilot	2	5.37

There is another approach called Lin's approach. So, the basic idea is that 2 different words are likely to have similar meanings if they occur in identical local context. Two different words are likely to have similar meaning if they occur in identical local context. So, if you take the word the facility will employ 500 new employees. The senses of facility are installation, proficiency, adeptness, readiness, toilet or bathroom. And in this particular case sense the one of installation will be the winner sense for the word. The subjects of employee can be words which types are organisation then plant, company, industry. It can be unit, aerospace, memory device, pilot, and swan with different weightages as shown here.

(Refer Slide Time: 41:16)



SIMILARITY AND HYPERNYMY

$$sim(A,B) = \frac{\log(P(common(A,B)))}{\log(P(describe(A,B)))}$$

If A is a *"Hill"* and B is a *"Coast"* then the commonality between A and B is that *"A is a GeoForm and B is a GeoForm"*.

$$sim(Hill, Coast) = \frac{2 * \log(P(GeoForm))}{\log(P(Hill)) + \log(P(Coast))}$$

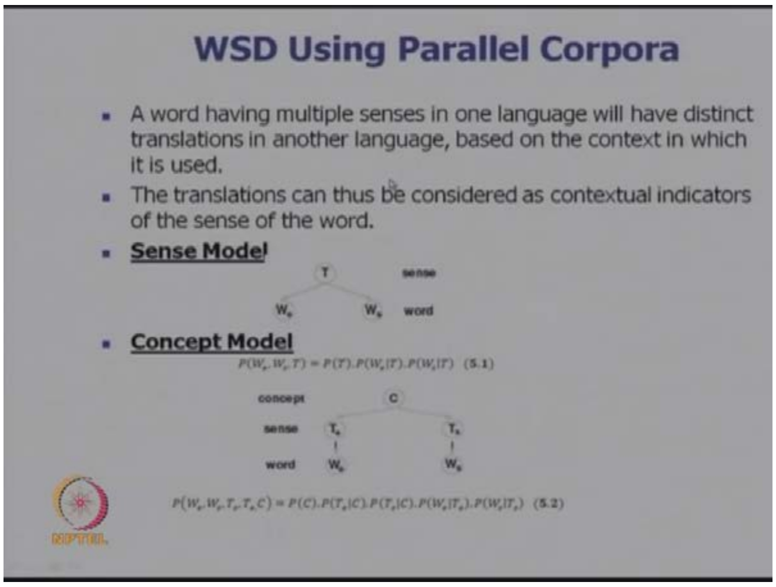
In general, *similarity* is directly proportional to the probability that the two words have the same super class (*Hypernym*)

entity 0.395
inanimate-object 0.167
natural-object 0.0163
geological-formation 0.00176
0.000113 natural-elevation
shore 0.0000836
0.0000189 hill
coast 0.0000216

To maximize similarity select that sense which has the same hypernym as most of the Selector words.

The similarity is computed between the target word and the words in the sense cluster So, if a is hill and b is coast then the commonality between A and B is that A is a geoform and A is a geoform. So, hypernymy is used for computing the similarity between a word and a sense component where it belongs. So, the words from the sense component to taken and similarity is computed between the target word and the context words. So, this gives rise to the determination of the sense.

(Refer Slide Time: 41:59)



WSD Using Parallel Corpora

- A word having multiple senses in one language will have distinct translations in another language, based on the context in which it is used.
- The translations can thus be considered as contextual indicators of the sense of the word.
- **Sense Model**

T
sense
W_e
W_s
word

- **Concept Model**

$$P(W_e, W_s, T) = P(T).P(W_e|T).P(W_s|T) \quad (5.1)$$

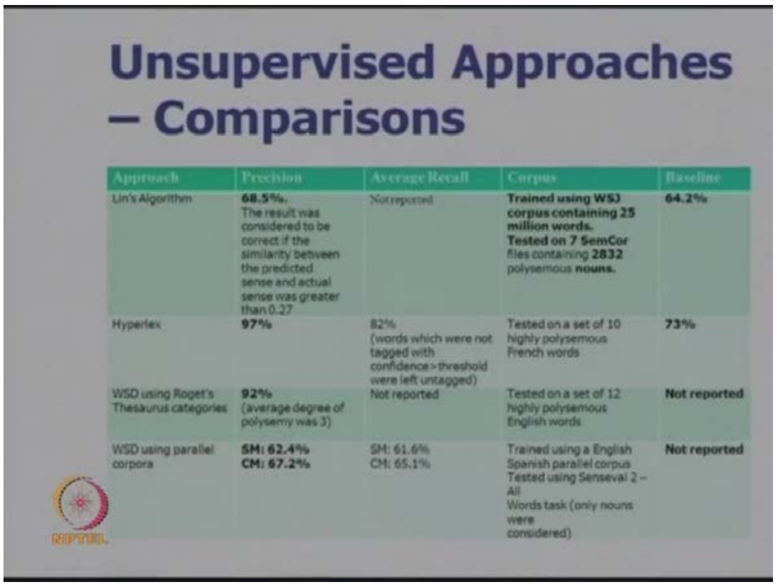
concept
C
sense
T_e
T_s
word
W_e
W_s

$$P(W_e, W_s, T_e, T_s, C) = P(C).P(T_e|C).P(T_s|C).P(W_e|T_e).P(W_s|T_s) \quad (5.2)$$

Word sense disambiguation using parallel corpora has become quite popular these days, because of the advent of the field of statistical machine translation. Large amount of parallel corpus is getting created and these parallel corpora are good for determining the sense of the word. So, the contextual word produces the clues and the sense label comes from the parallel aligned sentence. So, the sense of a sentence can be obtained given the aligned sentences. So, here is the description a word having multiple senses in one language will have distinct translations in another language this is a very powerful clue based on the context in which it is used. The translation can thus be considered as contextual indicators of the sense of the word.

\So, the there is a probabilistic formulation for this the sense is T its manifestation is word e, W e lets say in English and word s in another language s. So, probability of W e W s and T, this is a joint probability; this can be written as probability of T into probability of W e given T into probability of W s given T using the independence assumption. So, W e and W s do not have any dependence on each other. So, that term does not appear here. So, probability of a concept if for example, we use conceptual category from Roget's thesaurus then the formula will change this way on the joint probability. So, what is the use of this? The use of this is that we can obtain the parallel corpora and from the parallel corpora it is possible to get the sense of the word as a translation. And this is guided by probabilistic calculations which were shown on the slide.

(Refer Slide Time: 44:19)



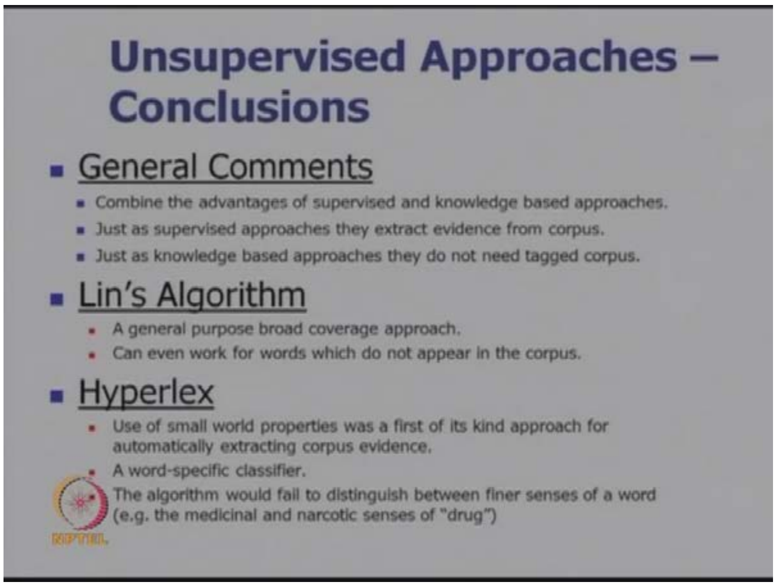
Unsupervised Approaches – Comparisons

Approach	Precision	Average Recall	Corpus	Baseline
Lin's Algorithm	**68.5%.** The result was considered to be correct if the similarity between the predicted sense and actual sense was greater than 0.27	Not reported	**Trained using WSJ corpus containing 25 million words. Tested on 7 SemCor** files containing **2832** polysemous **nouns.**	**64.2%**
Hyperlex	**97%**	82% (words which were not tagged with confidence>threshold were left untagged)	Tested on a set of 10 highly polysemous French words	**73%**
WSD using Roget's Thesaurus categories	**92%** (average degree of polysemy was 3)	Not reported	Tested on a set of 12 highly polysemous English words	**Not reported**
WSD using parallel corpora	**SM: 62.4%** **CM: 67.2%**	SM: 61.6% CM: 65.1%	Trained using a English Spanish parallel corpus Tested using Senseval 2 – All Words task (only nouns were considered)	**Not reported**

So, some remarks on supervised approaches to word sense disambiguation. We see that if the approach is Lin's algorithm; we get a precision of about 68.5 percent. The result was considered to be correct if the similarity between the predicted sense and the actual sense will be greater than 0.27. Recall is not reported, corpus is trained using wall street journal corpus containing 25 million words tested on 7 semcor files containing 2832 polysemous nouns the baseline accuracy is 68.4 percent. So, there is a four percent improvement. Hyperlex algorithm has a high accuracy about 90.7 percent, recall 82 percent tested on a set of 10 highly polysemous French words, baseline accuracy 73 percent. So, there is about 25 percent there is an accuracy improvement of about 25 points.

Yarowsky's algorithm for WSD using Roget's thesaurus, the accuracy is 92 percent tested on a 12 highly polysemous English words. A baseline accuracy is written as not reported, but it would be easy to calculate, because this comes from first sense of the word sense disambiguation using parallel corpora. We have here the accuracy values 62 to percents 67 percent when the senses are concerned and when the concepts are concerned. It is trained using in English Spanish parallel corpus tested using senseval2 all words task baseline is not reported.

(Refer Slide Time: 46:05)



Unsupervised Approaches – Conclusions

- General Comments
  - Combine the advantages of supervised and knowledge based approaches.
  - Just as supervised approaches they extract evidence from corpus.
  - Just as knowledge based approaches they do not need tagged corpus.
- Lin's Algorithm
  - A general purpose broad coverage approach.
  - Can even work for words which do not appear in the corpus.
- Hyperlex
  - Use of small world properties was a first of its kind approach for automatically extracting corpus evidence.
  - A word-specific classifier.
  - The algorithm would fail to distinguish between finer senses of a word (e.g. the medicinal and narcotic senses of "drug")

So, in general we can say that unsupervised word sense disambiguation algorithm has lot of interest in the community, because sense mark-up corpora is so difficult to obtain. But these algorithms are difficult in terms of high in terms of achieving high accuracy and in terms of producing the labels. But this would be good for languages whose resources are scarce.