Semantic Vector Space Models

Natalia Levshina ©2017

Summer School of Linguistics

Litomyšl, August 2017

Outline

1. Semantic Vector Space Models: introduction

2. Case study: classification of verbs depending on their subcategorization frames

Semantic Vector Space Models

- Main inspiration:
  - "You shall know the word by the company it keeps" (Firth 1957)
  - Words that occur in the same contexts tend to have similar meaning (Harris 1959)
- Origins in Computational Linguistics (cf. Deerwester et al. 1990, Schütze 1992, Lin 1998)
- Various applications in lexical and constructional corpus-based semantics (e.g. Levshina & Heylen 2014; Perek 2016)

A compact overview for linguists

- See my paper with Kris Heylen (Levshina & Heylen 2014)

The main steps

1. Create the vectors of co-occurrence frequencies of target words and contextual features (other words, subcategorization frames, etc.).

2. Normalize the frequencies (usually by transforming them into Poinwise Mutual Information scores), so that more surprising frequencies get more weight.

3. Compute the distances between the vectors (usually using the cosine measure).

4. Perform cluster analysis, MDS or use another exploratory technique.

What are contextual features?

- Bag-of-words models:
  - Take all words that occur within a certain window around a word (e.g. 5 words on the left and 5 words on the right).

-5	-4	-3	-2	-1	0	1	2	3	4	5
I	wandered	lonely	as	a	**cloud**	that	floats	on	high	ov’r

  - Large window: topic-related information

Topic: hospital

What are contextual features like?

- Bag-of-words models:
  - Take all words that occur within a certain window around a word (e.g. 5 words on the left and 5 words on the right).

-5	-4	-3	-2	-1	0	1	2	3	4	5
I	wandered	lonely	as	a	**cloud**	that	floats	on	high	ov'r

  - Small window: semantic relatedness

Some semantic relationships

- Synonyms, e.g. *begin – start*

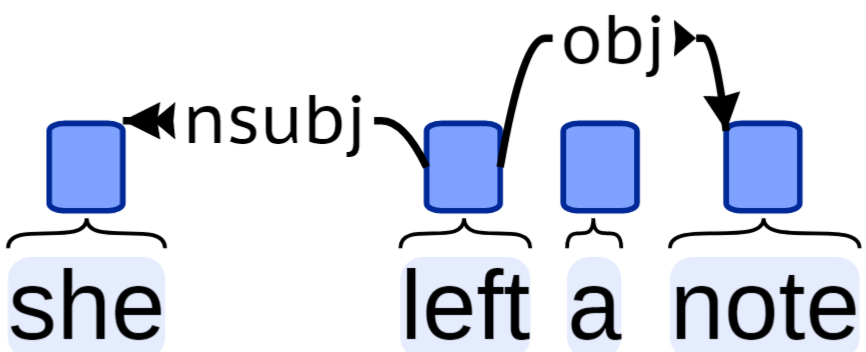
- Co-hyponyms, e.g. *fox terrier – beagle*

- Hyper- and hyponyms, e.g. *dog - beagle*

Images from Wikipedia

What are contextual features like?

- Syntactically enriched models:
  - Take into account the syntactic dependencies between the targets and other lexemes

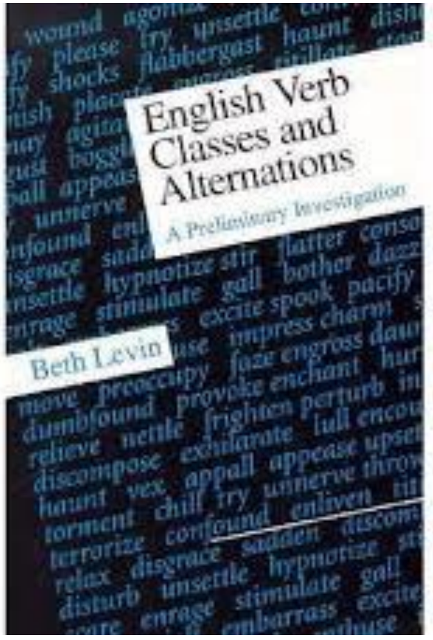


  - Good for capturing semantic similarity.

Image from http://universaldependencies.org/u/overview/simple-syntax.html

What are contextual features like?

- Fully syntactic information
  - E.g. subcategorisation frames for verbs, such as Subj_V_Indirect Object_Direct Object.
- Yields classes similar to Levin's (1993)



Of cabbages and kings (COCA)

	vegetable.n	grow.v	crown.n	royal.adj	make.v	…
cabbage	76	43	3	1	109	…
king	11	90	134	236	1122	…
queen	3	92	56	127	312	…
…	…	…	…	…	…	…

Of cabbages and kings (COCA)

	vegetable.n	**grow.v**	**crown.n**	**royal.adj**	**make.v**	**…**
cabbage	76	43	3	1	**109**	**…**
king	11	90	134	236	**1122**	**…**
queen	3	92	56	127	**312**	**…**
…	…	…	…	…	…	…

Weighting: PMI

- Pointwise Mutual Information

$$\text{PMI(target, context)} = \log_2 \frac{\text{P (target, word)}}{\text{P (target) * P (word)}}$$

Can be expressed as log (Observed / Expected) in a contingency table (Evert 2005).

- Negative values are replaced with zero.
- As a result of this weighting, the co-occurrence frequencies with highly frequent words (e.g. *like*) are less important, and infrequent events become more prominent.

Positive PMI

	vegetable.n	grow.v	crown.n	royal.adj	make.v	…
cabbage	76	43	3	1	**109**	…
king	11	90	134	236	**1122**	…
queen	3	92	56	127	**312**	…
…	…	…	…	…	…	…

	vegetable.n	grow.v	crown.n	royal.adj	make.v	…
cabbage	3.14	0.99	0	0	**0**	…
king	0	0	0.07	0	**0.14**	…
queen	0	0.74	0.25	0.51	**0**	…
…	…	…	…	…	…	…

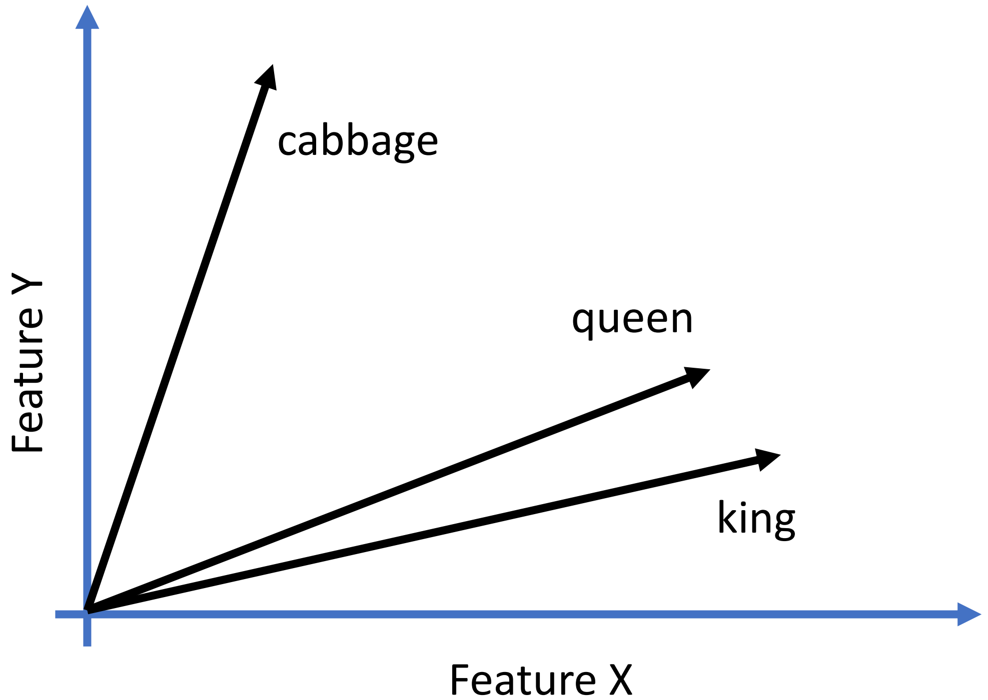
Why vector space?

- We speak about distributional vectors (sequences of PMI scores).

	vegetable.n	grow.v	crown.n	royal.adj	make.v	…
cabbage	3.14	0.99	0	0	0	…
king	0	0	0.07	0	0.14	…
queen	0	0.74	0.25	0.51	0	…
…	…	…	…	…	…	…

Why vector space?

- The weighted frequencies can be seen as coordinates defining the position of a target word in a multidimensional semantic feature space.



What is the cosine measure of similarity?

- If the angle between two vectors is small, the cosine will be large (up to 1), e.g. king and queen.
- If the angle is large, the cosine will be small (around 0), e.g. cabbage and king.
- **The greater the cosine of the angle between two vectors, the more similar the corresponding words.**

Outline

1. Semantic Vector Space Models: introduction

2. Case study: classification of verbs depending on their subcategorization frames

Data

- Universal Dependencies Corpus
  - http://universaldependencies.org/
- English subcorpus (254K): blogs, social media, reviews
- > 2K verbs (types)
- 28 subcategorization frames with various arguments and adjuncts, e.g.
  - Subject + Verb + Adverbial Modifier (e.g. John runs fast.)
  - Subject + Verb + Indirect Object + Direct Object (e.g. John gives Mary a book).

Data set verbframes

```
> dim(verbframes)
[1] 598  28
> verbframes[1:8, 1:8]
```

	Intr1	Intr2	Intr3	Intr4	Intr5	Intr6	Intr8	Trans1
come	20	22	21	64	67	48	26	2
replace	0	6	1	0	0	0	0	4
retire	2	6	0	0	0	2	1	0
attack	0	1	2	0	0	0	0	7
launch	0	2	2	0	0	0	0	2
kill	0	4	7	0	0	0	1	13
have	188	28	2	14	71	4	5	586
found	2	1	2	0	0	0	1	0

Weighting the frequencies

1. Transform frequencies into Pointwise Mutual Information scores

```
> verbframes.exp <-
chisq.test(verbframes)$expected
```

```
> verbframes.pmi <-
log2(verbframes/verbframes.exp)
```

2. Transform PMI into Positive PMI

```
> verbframes.ppmi <- verbframes.pmi
```

```
> verbframes.ppmi[verbframes.ppmi < 0] <- 0
```

Selecting a smaller list of verbs

```
> verblist <- c("give" ,"send", "offer","hand",
"create", "build", "produce", "jump", "come",
"go", "walk", "tell", "say", "speak", "talk",
"communicate","think","believe","assume",
"understand", "discuss", "expect", "see", "hear",
"feel",  "watch", "listen", "start", "begin",
"finish")

> verbframes.short <-
verbframes.ppmi[rownames(verbframes) %in%
verblist,]
```

Computing the cosine measures

- Function cosine from add-on package lsa. Note that we need to transpose the data (i.e. swap the rows and columns).

```
> library(lsa)
> verbframes.cos <- cosine(t(verbframes.short))
```

- We'll assign zeros to diagonal elements (similarity of a word to itself), for convenience:

```
> diag(verbframes.cos) <- 0
```

Which verbs are the most similar?

```
> max(verbframes.cos)
[1] 0.9557553

> which(verbframes.cos == max(verbframes.cos),
arr.ind = TRUE)
      row col
walk  27  11
go    11  27
```

Which verbs are similar to give?

```
> verbframes.cos[rownames(verbframes.cos) ==
"give", ]
```

[output omitted]

For convenience, round the numbers up to three decimal points:

```
> round(verbframes.cos[rownames(verbframes.cos) ==
"give", ], 3)
```

[output omitted]

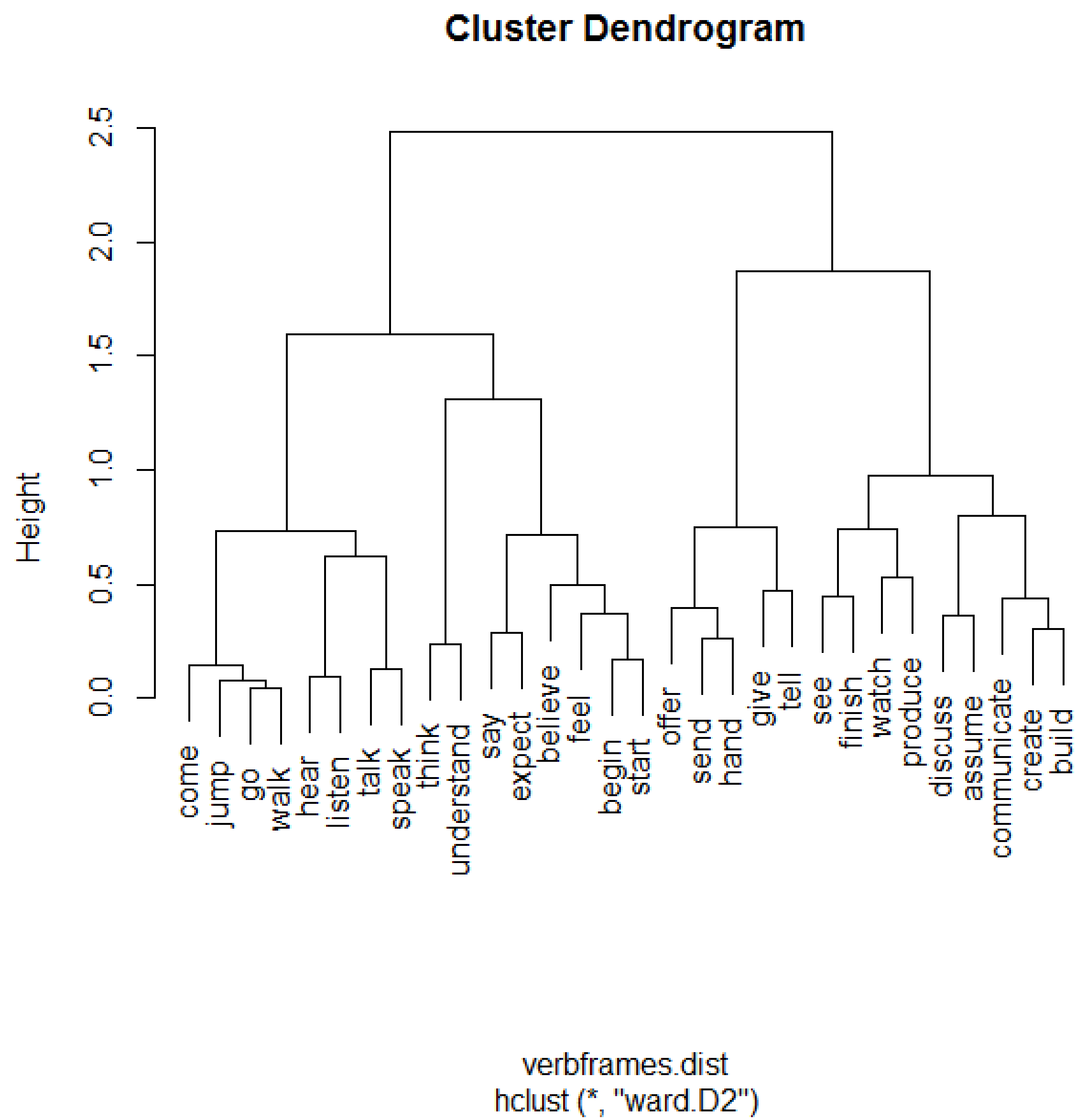
Hierarchical cluster analysis

1. Transform the similarity scores into distances by subtracting the cosines from 1:

```
> verbframes.dist <- as.dist(1 - verbframes.cos)
```

2. Perform HCA (see previous lecture):

```
> verbframes.clust <- hclust(verbframes.dist,
method = "ward.D2")
> plot(verbframes.clust)
```



Cluster Dendrogram
Height
2.5
2.0
1.5
1.0
0.5
0.0
come
jump
go
walk
hear
listen
talk
speak
think
understand
say
expect
believe
feel
begin
start
offer
send
hand
give
tell
see
finish
watch
produce
discuss
assume
communicate
create
build
verbframes.dist
hclust (*, "ward.D2")

Comparing properties of clusters

```
> verbs1 <- c("come", "go", "walk", "jump")
> verbs2 <- c("hear", "listen", "talk", "speak")
> cluster1 <-
verbframes.short[rownames(verbframes.short) %in%
verbs1,]
> cluster2 <-
verbframes.short[rownames(verbframes.short) %in%
verbs2,]
```

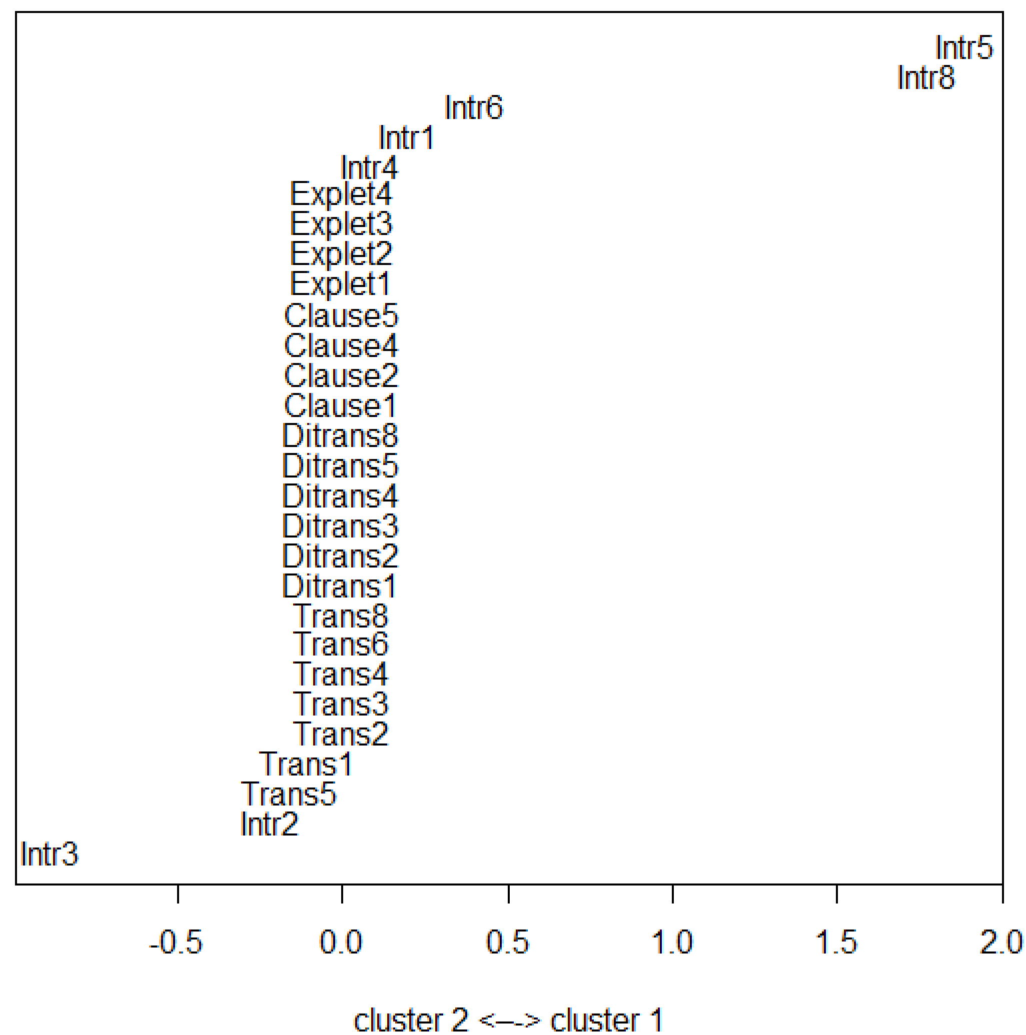
Compute the mean PMIs and the differences between them:

```
> cluster1.means <- colMeans(cluster1)
> cluster2.means <- colMeans(cluster2)
> diff <- cluster1.means - cluster2.means
```

Making a “snake plot”

```
> plot(sort(diff), 1:length(diff), type = "n",
xlab = "cluster 2 <--> cluster 1", yaxt = "n",
ylab = "")

> text(sort(diff), 1:length(diff),
names(sort(diff)))
```



Intr5
Intr8
Intr6
Intr1
Intr4
Explet4
Explet3
Explet2
Explet1
Clause5
Clause4
Clause2
Clause1
Ditrans8
Ditrans5
Ditrans4
Ditrans3
Ditrans2
Ditrans1
Trans8
Trans6
Trans4
Trans3
Trans2
Trans1
Trans5
Intr2
Intr3
-0.5
0.0
0.5
1.0
1.5
2.0
cluster 2 <–-> cluster 1

Distinctive subcategorization frames

- Frame Intr3: V + Oblique
  - Talk **to** your academic **adviser**, see what they recommend.
- Frame Intr5: Subj + V + AdvMod
  - **Now** comes the fun part.
- Frame Intr8: V + AdvMod
  - Coming **soon**!

Divisive clustering

- Splits the data into a pre-defined number of clusters
- Popular options: k-means, partitioning around medoids (more robust)

Divisive clustering with PAM

```
> library(cluster)
> verbframes.pam <- pam(verbframes.dist, k = 3)
> verbframes.pam$clustering
      come      hear    create      give
         1         1         2         3
       say     think    expect   believe
         1         1         2         1
      send      tell        go       see
         3         3         1         2
     build      talk     begin    finish
         2         1         1         2
```

…

Conclusions

- Most relationships are interpretable semantically. Given the small corpus size, the results are surprisingly good (some people recommend > 50M for SVS).
- The hierarchical and divisive solutions are very similar.

References

- Deerwester, S., Dumais, S.T., Furnas, G.W., Landayer, T.K., & Harshman, R. (1990). Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science 41:* 391–407.
- Everitt, B.S., Landau, S., Leese, M., & Stahl, D. (2011). *Cluster Analysis* (5th ed.). Chichester: Wiley.
- Firth, J.R. (1957). A synopsis of linguistic theory 1930–1955. In J. R. Firth (Ed.), *Studies in Linguistic Analysis,* 1–32) Oxford: Blackwell.
- Harris, Z. (1954). Distributional structure. *Word 10*(2/3): 146–162.
- Levshina, N., & Heylen, K. (2014). A radically data-driven construction grammar: Experiments with Dutch causative constructions. In R. Boogaart, T. Colleman, & G. Rutten (Eds.), *Extending the Scope of Construction Grammar,* 17–46. Berlin/New York: Mouton de Gruyter.
- Lin, D. (1998). Automatic retrieval and clustering of similar words. *Proceedings of the 17th International Conference on Computational linguistics*, Montreal, Canada, August 1998, 768–774.
- Perek, F. (2016). Using distributional semantics to study syntactic productivity in diachrony: A case study. *Linguistics* 54(1): 149–188.
- Schütze, H. (1992). Dimensions of meaning. In *Proceedings of Supercomputing 92,* 787–796. Minneapolis, MN.