TEXT CLASSIFICATION

Intent, topic, sentiment, etc.

Wongnai Challenge

- Predict star rating from review text

output

1 check-in

ร้องเรียน

ก๊วยเตี๋ยวอร่อย ราคาถูก นั่งทานในห้องแอร์

เมนูเด็ด: บะหมี่ต้มยำหมูแดง

input

ส่วนตัวชอบก๊วยเตี๋ยวต้มยำมากค่ะ รสชาติอร่อยไม่ต้องปรุงเลย แต่ซุปเปอร์ตีนไก่ใส่ถั่วงอกมาด้วย เหมือนเป็นเกาเหลาตีนไก่มากกว่าซุปเปอร์ตีนไก่ รสชาติก็ยังไม่กลมกล่อมเท่าก๊วยเตี๋ยวต้มยำ ตีนไก่เปื่อยดี ทานง่าย

Duyu Tang Document Modeling with Gated Recurrent Neural Network for Sentiment Classification, 2015 http://aclweb.org/anthology/D15-1167

Wongnai challenge top model Accuracy 0.5844

Thai2fit (contextualized word embedding with adaptation) Accuracy 0.60925

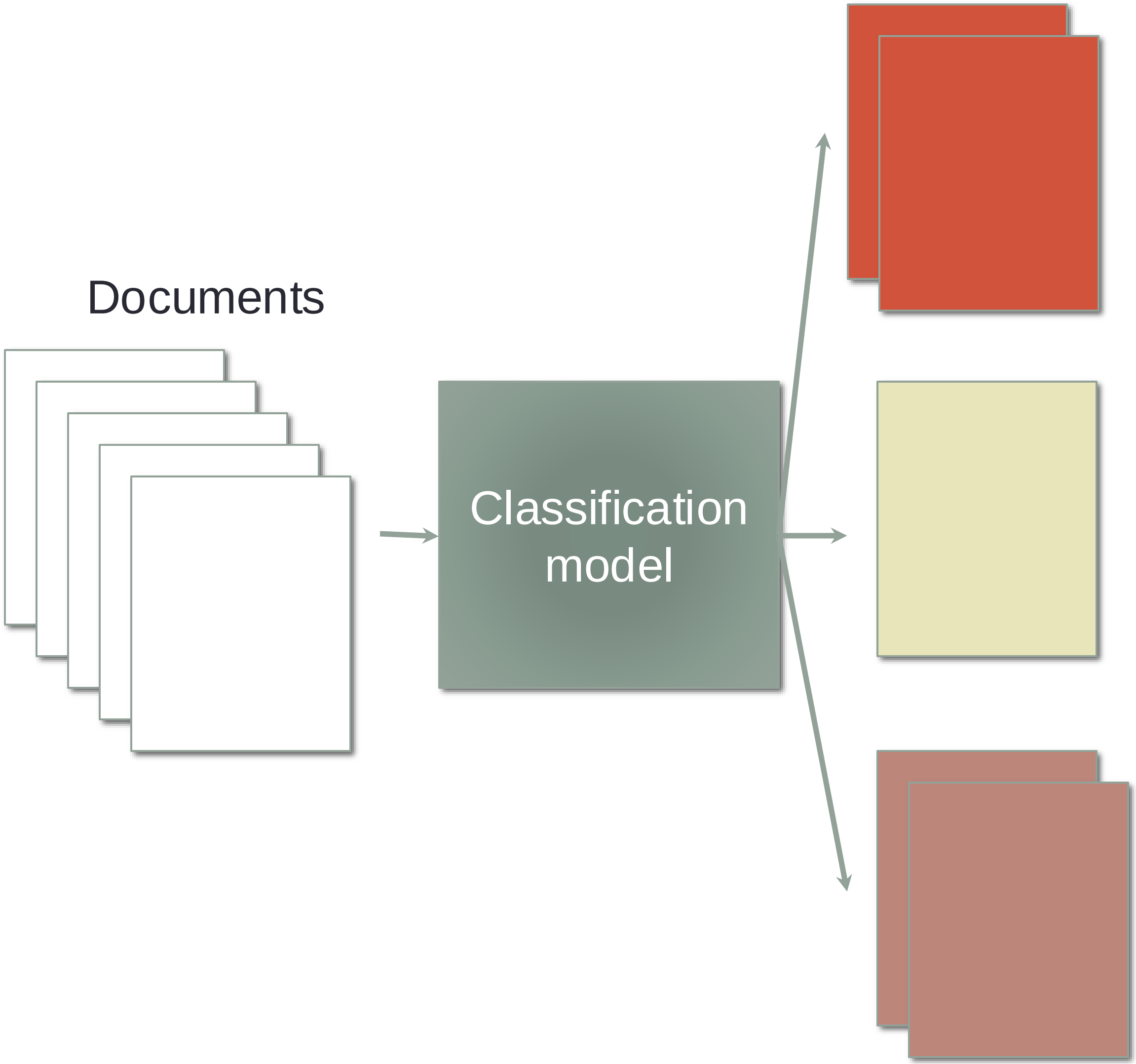
Yelp reviews

Corpus	#docs	#s/d	#w/d	$\|V\|$	#class	Class Distribution
Yelp 2013	335,018	8.90	151.6	211,245	5	.09/.09/.14/.33/.36
Yelp 2014	1,125,457	9.22	156.9	476,191	5	.10/.09/.15/.30/.36
Yelp 2015	1,569,264	8.97	151.9	612,636	5	.10/.09/.14/.30/.37
IMDB	348,415	14.02	325.6	115,831	10	.07/.04/.05/.05/.08/.11/.15/.17/.12/.18

	Yelp 2013		Yelp 2014		Yelp 2015		IMDB	
	Accuracy	MSE	Accuracy	MSE	Accuracy	MSE	Accuracy	MSE
Majority	0.356	3.06	0.361	3.28	0.369	3.30	0.179	17.46
SVM + Unigrams	0.589	0.79	0.600	0.78	0.611	0.75	0.399	4.23
SVM + Bigrams	0.576	0.75	0.616	0.65	0.624	0.63	0.409	3.74
SVM + TextFeatures	0.598	0.68	0.618	0.63	0.624	0.60	0.405	3.56
SVM + AverageSG	0.543	1.11	0.557	1.08	0.568	1.04	0.319	5.57
SVM + SSWE	0.535	1.12	0.543	1.13	0.554	1.11	0.262	9.16
JMARS	N/A	–	N/A	–	N/A	–	N/A	4.97
Paragraph Vector	0.577	0.86	0.592	0.70	0.605	0.61	0.341	4.69
Convolutional NN	0.597	0.76	0.610	0.68	0.615	0.68	0.376	3.30
Conv-GRNN	0.637	0.56	0.655	0.51	0.660	0.50	0.425	**2.71**
LSTM-GRNN	**0.651**	**0.50**	**0.671**	**0.48**	**0.676**	**0.49**	**0.453**	3.00

Attention (both levels)	**68.6**

Text/document classification



Document classification

Type	Focus	Example
Topic	Subject matter	Sports vs Technology
Sentiment/opinion	Emotion (current state)	Negative vs Positive
Intent	Action (future state)	Order vs Inquiry

คืนนี้จะได้ดูตอนใหม่แล้ว #ออเจ้า #อดใจไม่ไหว

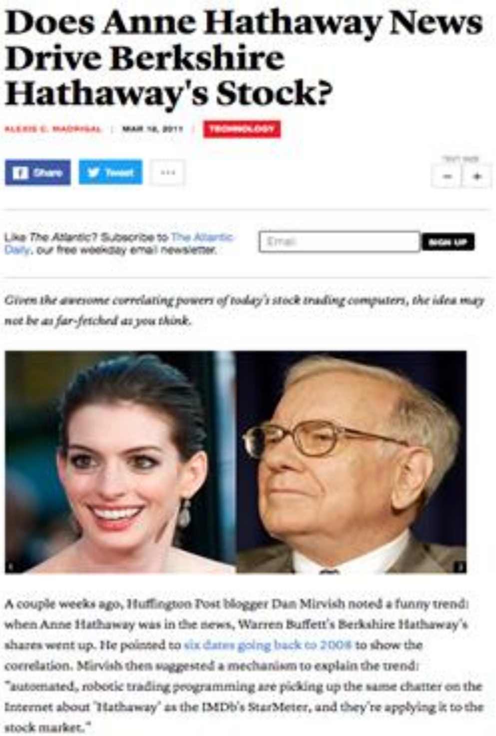
Topic: บุพเพสันนิวาส

Sentiment: positive

Action: watch

อยากจะสั่งพิซซ่าหน้าฮาวายเอี่ยนหน่อยครับ

Action: order_hawaiian



Does Anne Hathaway News Drive Berkshire Hathaway's Stock?

Like *The Atlantic*? Subscribe to The Atlantic Daily, our free weekday email newsletter.

Given the awesome correlating powers of today's stock trading computers, the idea may not be as far-fetched as you think.

A couple weeks ago, Huffington Post blogger Dan Mirvish noted a funny trend: when Anne Hathaway was in the news, Warren Buffett's Berkshire Hathaway's shares went up. He pointed to six dates going back to 2008 to show the correlation. Mirvish then suggested a mechanism to explain the trend: "automated, robotic trading programming are picking up the same chatter on the Internet about 'Hathaway' as the IMDb's StarMeter, and they're applying it to the stock market."

https://www.theatlantic.com/technology/archive/2011/03/does-anne-hathaway-news-drive-berkshire-hathaways-stock/72661/

Other classification applications

- Spam filtering
- Authorship id
- Auto tagging (information retrieval)
- Trend analysis

Text classification definition

- Input
  - Set of documents: $D = \{d_1, d_2, d_3, \ldots, d_M\}$
    - Each document is composed of words
      - $d1 = [w_{11}, w_{12}, \ldots w_{1N}]$
  - Set of classes: $C = \{c_1, c_2, c_3, .., c_J\}$

- Output
  - The predicted class c from the set C

Rule-based classification

- Rules based on phrases or other features
  - Wongnai Rating
    - แมลงสาบ -> 2 ดาว
    - อร่อย -> 4 ดาว
    - ไม่อร่อย -> 2 ดาว
    - …
  - What if the phrase is ไม่ค่อยอร่อย
  - New rule
    - อร่อย โดยที่ไม่มีคำว่าไม่อยู่แถวๆนั้น -> 4 ดาว
  - What if the phrase is ไม่ถูกแต่อร่อย
- This can yield very good results but…
- Building and maintaining rules is expensive!
- Or keep a word list of positive and negative words

Supervised text classification definition

- Input
  - Set of documents: $D = \{d_1, d_2, d_3, \ldots, d_M\}$
  - And labels $Y = \{y_1, y_2, y_3, \ldots, y_M\}$
    - Each document is composed of words
      - $d1 = [w_{11}, w_{12}, \ldots w_{1N}]$
  - Set of classes: $C = \{c_1, c_2, c_3, .., c_J\}$
- Output
  - A classifier $H: d \rightarrow c$

What classifier?

- Any classifier you like
- k-NN
- Naïve Bayes
- Logistic regression
- SVM
- Neural networks

We use this kind of classifier before in the previous homework

Outline

- Naïve Bayes
- Neural methods
- Topic Models
  - Latent topic models (LDA)

Bag of words representation

1 check-in

ร้องเรียน

ก๋วยเตี๋ยวอร่อย ราคาถูก นั่งทานในห้องแอร์

เมนูเด็ด: บะหมี่ต้มยำหมูแดง

ส่วนตัวชอบก๋วยเตี๋ยวต้มยำมากค่ะ รสชาติอร่อยไม่ต้องปรุงเลย แต่ซุปเปอร์ตีนไก่ใส่ถั่วงอกมาด้วย เหมือนเป็นเกาเหลาตีนไก่มากกว่าซุปเปอร์ตีนไก่ รสชาติก็ยังไม่กลมกล่อมเท่าก๋วยเตี๋ยวต้มยำ ตีนไก่เปื่อยดี ทานง่าย

H [ส่วนตัวชอบก๋วยเตี๋ยวต้มยำมากค่ะ รสชาติอร่อยไม่ต้องปรุงเลย แต่ซุปเปอร์ตีนไก่ใส่ถั่วงอกมาด้วย เหมือนเป็นเกาเหลาตีนไก่มากกว่าซุปเปอร์ตีนไก่ รสชาติก็ยังไม่กลมกล่อมเท่าก๋วยเตี๋ยวต้มยำ ตีนไก่เปื่อยดี ทานง่าย] = 3

Bag of words representation

H [ส่วนตัวชอบก๋วยเตี๋ยวต้มยำมากค่ะ รสชาติอร่อยไม่ต้องปรุงเลย แต่ซุปเปอร์ตีนไก่ใส่ถั่วงอกมาด้วย เหมือนเป็นเกาเหลาตีนไก่มากกว่าซุปเปอร์ตีนไก่ รสชาติก็ยังไม่กลมกล่อมเท่าก๋วยเตี๋ยวต้มยำ ตีนไก่เปื่อยดี ทานง่าย] = 3

Bag of words only care about the presence of words or features but ignore word position and context

H [ชอบ, อร่อย, ไม่, ไม่, กลมกล่อม, ทานง่าย] = 3

Bag of words representation

H [ส่วนตัวชอบก๊วยเตี๋ยวต้มยำมากค่ะ รสชาติอร่อยไม่ต้องปรุงเลย แต่ซุปเปอร์ตีนไก่ใส่ถั่วงอกมาด้วย เหมือนเป็น
เกาเหลาตีนไก่มากกว่าซุปเปอร์ตีนไก่ รสชาติก็ยังไม่กลมกล่อมเท่าก๊วยเตี๋ยวต้มยำ ตีนไก่เปื่อยดี ทานง่าย] = 3

Bag of words only care about the presence of words or features but ignore word position and context

H [

Word	Count
ชอบ	1
อร่อย	1
ไม่	2
กลมกล่อม	1
ทานง่าย	1

] = 3

Bag of words for classification intuition

Test review

แมงสาบ
ใช้ได้
ถูก
อร่อย

1star

<u>แมงสาบ</u>
สกปรก
แย่

3star

<u>ใช้ได้</u>
<u>ถูก</u>
<u>อร่อย</u>
คาดฝัน

5star

เหาะ
เชฟ
<u>อร่อย</u>
ยอด

Bag of words for classification intuition

Figure 15.3 A spatial visualization of the document vectors for the four Shakespeare play documents, showing just two of the dimensions, corresponding to the words *battle* and *fool*. The comedies have high values for the *fool* dimension and low values for the *battle* dimension.

Reference: Jurafsky, Dan, and James H. Martin. Speech and language processing. 3rd edition draft, https://web.stanford.edu/~jurafsky/slp3/, August 2017

Bayes' Rule for classification

- A simple classification model
- Given document d, find the class c
  - $\underset{c}{\text{Argmax}}\ P(c|d)$

$$=\underset{c}{\text{Argmax}}\ \frac{P(d|c)\,P(c)}{P(d)}$$

Bayes' Rule

$$=\underset{c}{\text{Argmax}}\ P(d|c)\,P(c)$$

P(d) is constant wrt to c

$$=\underset{c}{\text{Argmax}}\ P(x_1, x_2, \ldots, x_n \mid c)\,P(c)$$

The document is represented by features $x_1, x_2, \ldots, x_n$

Bayes' Rule for classification

- A simple classification model
- Given document d, find the class c
  - $\underset{c}{\text{Argmax}}\ P(c|d)$

$$= \underset{c}{\text{Argmax}}\ \frac{P(d|c)\,P(c)}{P(d)}$$

$$= \underset{c}{\text{Argmax}}\ P(d|c)\,P(c)$$

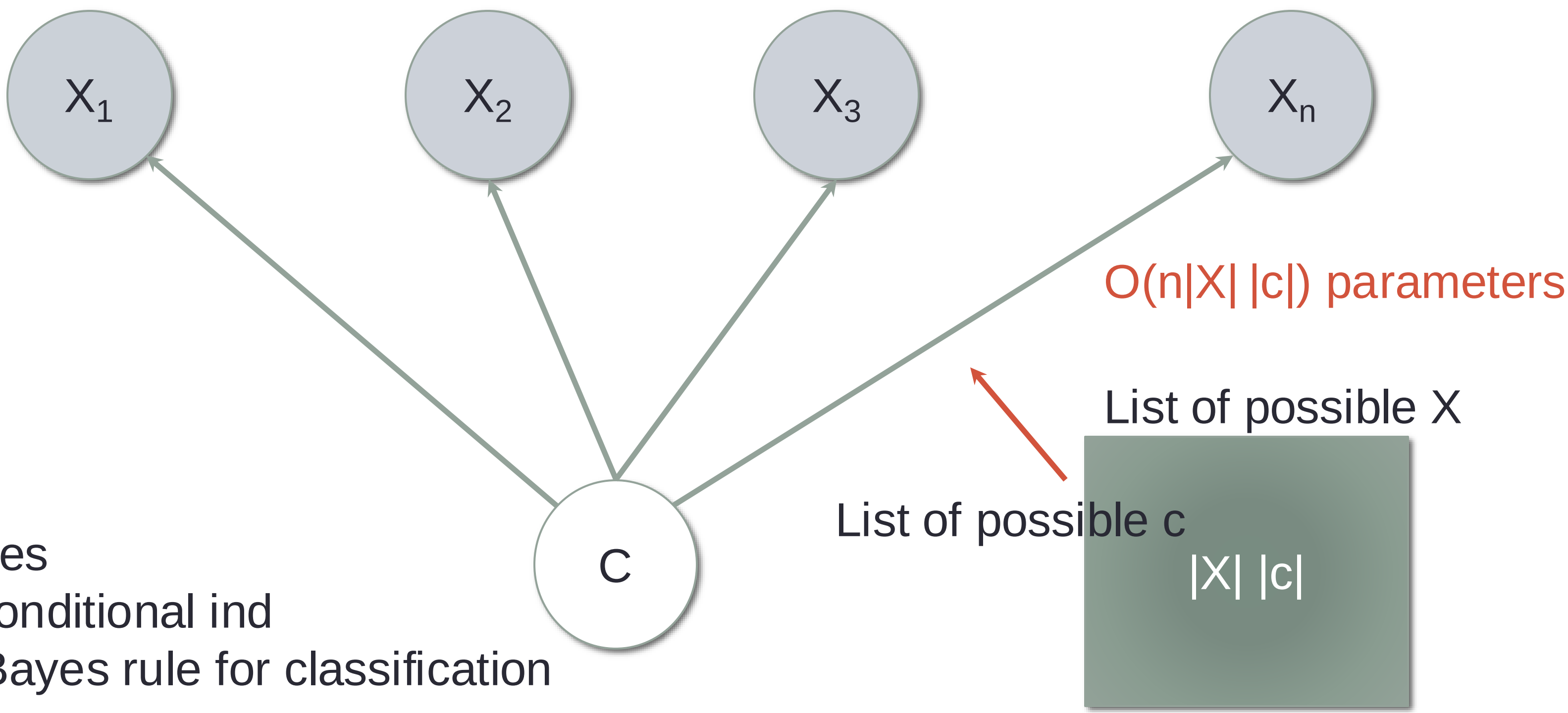
likelihood prior

$$= \underset{c}{\text{Argmax}}\ P(x_1, x_2, \ldots, x_n \mid c)\,P(c)$$

$P(x_1, x_2, \ldots, x_n \mid c)$ requires $O(|X|^n\,|c|)$ parameters. Cannot train

Bag of words assumption

- $P(x_1, x_2, \ldots, x_n \mid c)\, P(c) = P(x_1|c)\, P(x_2|c)\, P(x_3|c) \ldots P(x_n|c)\, P(c)$
  - Conditional independence



Naïve Bayes
Naïve – conditional ind
Bayes – Bayes rule for classification

Bags of words and NB

Probability of drawing words from the bag

Word	Distribution (class=1)
ชอบ	0.1
อร่อย	0.1
ไม่	0.5
กลมกล่อม	0.2
ทานง่าย	0.1

P(“ไม่อร่อยไม่ชอบ”| c = 1)

= P(ไม่ | c= 1) P(อร่อย | c= 1) P(ไม่ | c= 1) P(ชอบ | c= 1)

= 0.5 * 0.1 * 0.5* 0.1

Learning the Naïve Bayes model

- As usual counts
- P(x|c)

x_n is a feature that counts word occurrence

x_1 how many times ยอด appear

- $$P(x = \text{“ยอด”} \mid c=5) = \frac{\text{count}(x = \text{“ยอด”}, c = 5)}{\text{count}(c = 5)}$$

List of possible counts

List of classes

|X| |c|

- P(c)
- $$P(c = 5) = \frac{\text{count}(c = 5)}{\text{count (all reviews)}}$$

- This is the Maximum Likelihood Estimate (MLE)

Learning the Naïve Bayes model

- What if we encounter zeroes in our table
- P(x|c)

x_n is a feature that counts word occurrence

x_1 how many times ยอด appear

- P(x = “ยอด”| c=5) = count(x =“ยอด”, c = 5) / count(c = 5)

$$P(x = \text{“ยอด”} \mid c=5) = \frac{\text{count}(x = \text{“ยอด”},\ c = 5)}{\text{count}(\ c = 5)}$$

List of possible counts

List of classes

|X| |c|

- P(‘ร้าน นี้ ราด หน้า ยอด ผัก ไม่ อร่อย’|c = 2)

= P(x= “ร้าน”|c=2)*P(x= “นี้”|c=2)… P(x= “ยอด”|c=2) *...

= 0

Zero probability regardless of other words

One solution: add-1 smoothing (a hyperparameter to tune)

What about unknown words (OOV)? Drop them (no calculation)

Naives Bayes

- Can use other features beside word counts
  - Feature engineering – restaurant name, location, price range, reviewer id, date of review
  - Tedious but very powerful
    - Features > 10000
- Pros: very fast, very small model
- Need to remove stop words
- Robust especially for small training data (hand-crafted rules)
- A good fast baseline. Always try Naive Bayes or logistic regression in model search.
- Even with lots of data and rich features, Naives Bayes can be very competitive and fast!

Naive Bayes vs Logistic regression
Generative vs Discriminative modeling

Given data x, predict y

- Naïve Bayes are generative models

$$y^* = argmax_y \frac{P(x|y)P(y)}{P(x)}$$

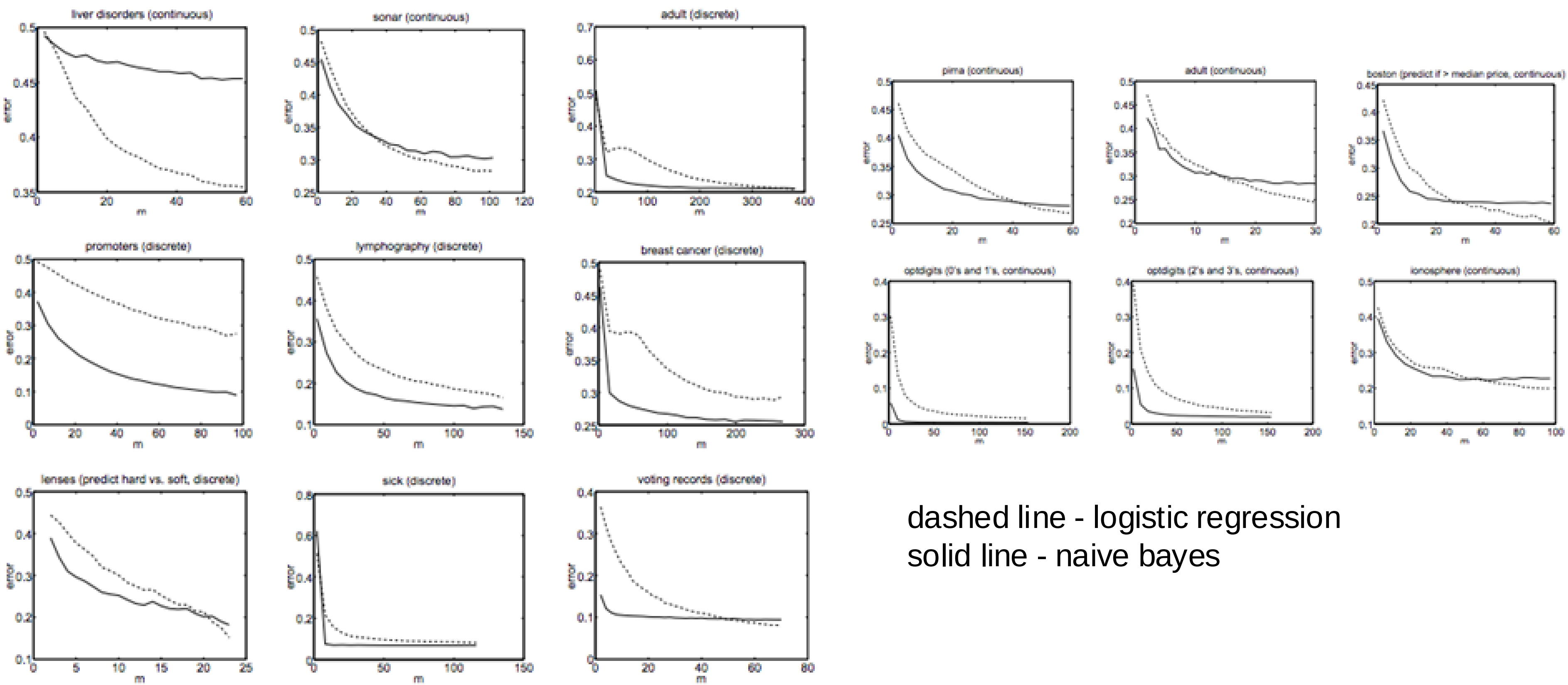
- Logistic regression are discriminative models
  - Note P(y|x) can be any function that outputs y given x (a neural network)

$$y^* = argmax_y P(y|x)$$

- Logistic regression and Naive Bayes are linear models (linear decision boundary)
- They are quite interchangeable.

Naive Bayes vs Logistic regression Generative vs Discriminative modeling

When training data is small, Naive Bayes performs better. When training data is large, Logistic regression performs better.



dashed line - logistic regression
solid line - naive bayes

http://ai.stanford.edu/~ang/papers/nips01-discriminativegenerative.pdf

Fast and good classification using n-grams

- Features: n-grams (bag of phrases)
- Model: logistic regression
- Very competitive results

Model	Yelp'13	Yelp'14	Yelp'15	IMDB
SVM+TF	59.8	61.8	62.4	40.5
CNN	59.7	61.0	61.5	37.5
Conv-GRNN	63.7	65.5	66.0	42.5
LSTM-GRNN	65.1	67.1	67.6	45.3
fastText	64.2	66.2	66.6	45.2

Table 3: Comparision with Tang et al. (2015). The hyper-parameters are chosen on the validation set. We report the test accuracy.

Bag of Tricks for Efficient Text Classification
https://arxiv.org/pdf/1607.01759.pdf

Fast and good classification using n-grams

- Features: n-grams (bag of phrases)
- Model: logistic regression
- Very competitive results

	Zhang and LeCun (2015)		Conneau et al. (2016)			fastText
	small char-CNN	big char-CNN	depth=9	depth=17	depth=29	$h = 10$, bigram
AG	1h	3h	24m	37m	51m	1s
Sogou	-	-	25m	41m	56m	7s
DBpedia	2h	5h	27m	44m	1h	2s
Yelp P.	-	-	28m	43m	1h09	3s
Yelp F.	-	-	29m	45m	1h12	4s
Yah. A.	8h	1d	1h	1h33	2h	5s
Amz. F.	2d	5d	2h45	4h20	7h	9s
Amz. P.	2d	5d	2h45	4h25	7h	10s

Table 2: Training time for a single epoch on sentiment analysis datasets compared to char-CNN and VDCNN.

Bag of Tricks for Efficient Text Classification
https://arxiv.org/pdf/1607.01759.pdf

Tag prediction

Model	prec@1	Running time	
		Train	Test
Freq. baseline	2.2	-	-
Tagspace, $h = 50$	30.1	3h8	6h
Tagspace, $h = 200$	35.6	5h32	15h
fastText, $h = 50$	31.2	6m40	48s
fastText, $h = 50$, bigram	36.7	7m47	50s
fastText, $h = 200$	41.1	10m34	1m29
fastText, $h = 200$, bigram	46.1	13m38	1m37

Table 5: Prec@1 on the test set for tag prediction on YFCC100M. We also report the training time and test time. Test time is reported for a single thread, while training uses 20 threads for both models.

Bag of Tricks for Efficient Text Classification
https://arxiv.org/pdf/1607.01759.pdf

Naïve Bayes tricks for text classification

- **Domain specific features**
- Count words after “not” as a different word
  - I don’t go there. -> I don’t go_not there_not
- Upweighting: double counting words at important locations
  - Words in titles
  - First sentence of each paragraph
  - Sentences that contain title words

Context-Sensitive Learning Methods for Text Categorization
https://www.researchgate.net/publication/2478208_Context-Sensitive_Learning_Methods_for_Text_Categorization

Information retrieval using location and category information
https://www.jstage.jst.go.jp/article/jnlp1994/7/2/7_2_141/_article

Automatic text categorization using the importance of sentences
https://dl.acm.org/citation.cfm?id=1072331

Different variants of Naive Bayes

- What we described was Multinomial Naive Bayes
  - Takes in word counts (Term frequency - TF)
  - Assumes length independent of class, TF follows Poisson dist
  - Can also take in a binary version of word counts
- There's also Multi-variate Bernoulli Naive Bayes
  - Takes in binary version of word counts
  - Slightly different assumptions, also consider probability when count = 0
- SVM-NB (SVM with NB as features)
- etc.

Additional readings

"Spam Filtering with Naive Bayes – Which Naive Bayes?"
http://www2.aueb.gr/users/ion/docs/ceas2006_paper.pdf

"Baselines and Bigrams: Simple, Good Sentiment and Topic Classification"

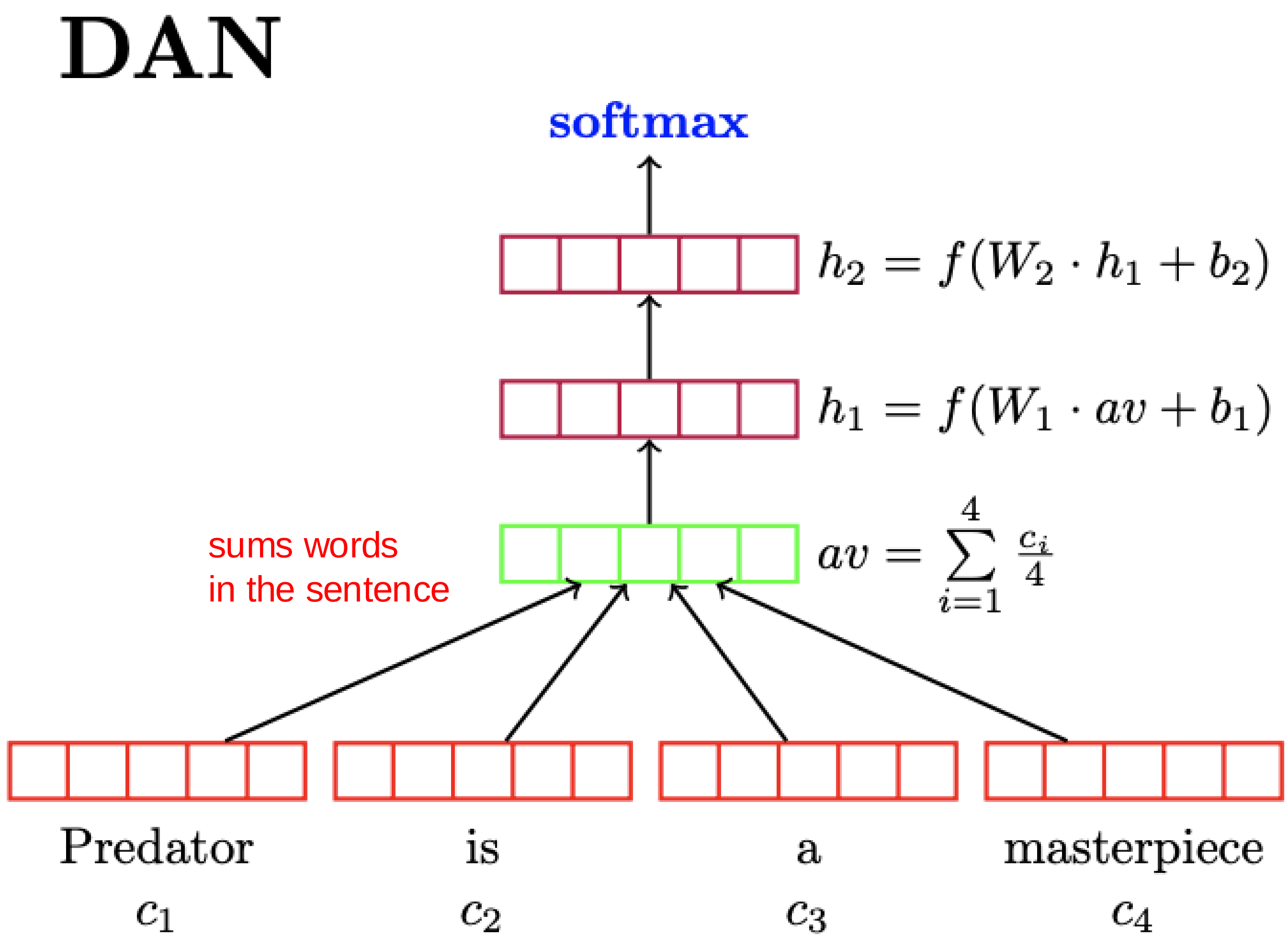
http://www.aclweb.org/anthology/P12-2018

https://www.kaggle.com/jhoward/nb-svm-strong-linear-baseline

Neural methods

- Sentence/document embedding
  - Deep Averaging Networks, USE, sentence embeddings

Deep Averaging Networks (DAN)



https://www.aclweb.org/anthology/P15-1162/ 2015

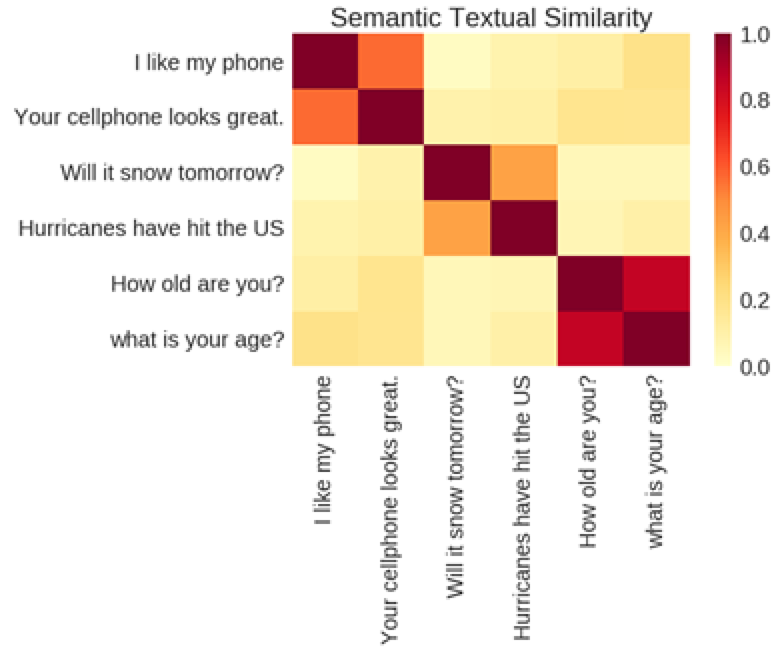
Universal Sentence Encoder (USE)

A model focusing on sentence representation

Use sentencepiece tokenization

Pre-trained then used anywhere

Based on (1) DAN (lite version) or (2) Transformer



Official implementation with pretrained weights

https://tfhub.dev/google/collections/universal-sentence-encoder/1

https://ai.googleblog.com/2018/05/advances-in-semantic-textual-similarity.html

Final words on text classification

Current state-of-the-art are about learning representations

Unsupervised pre-training of text (Word2Vec, BERT, ULMFit, simCSE, ConGen, etc)

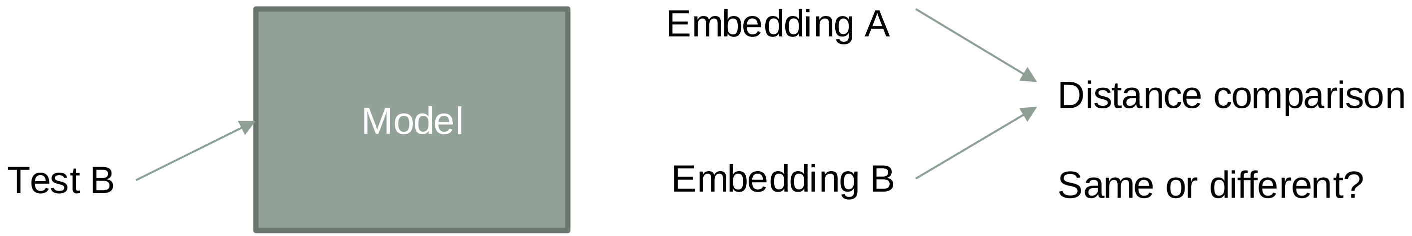
Trend

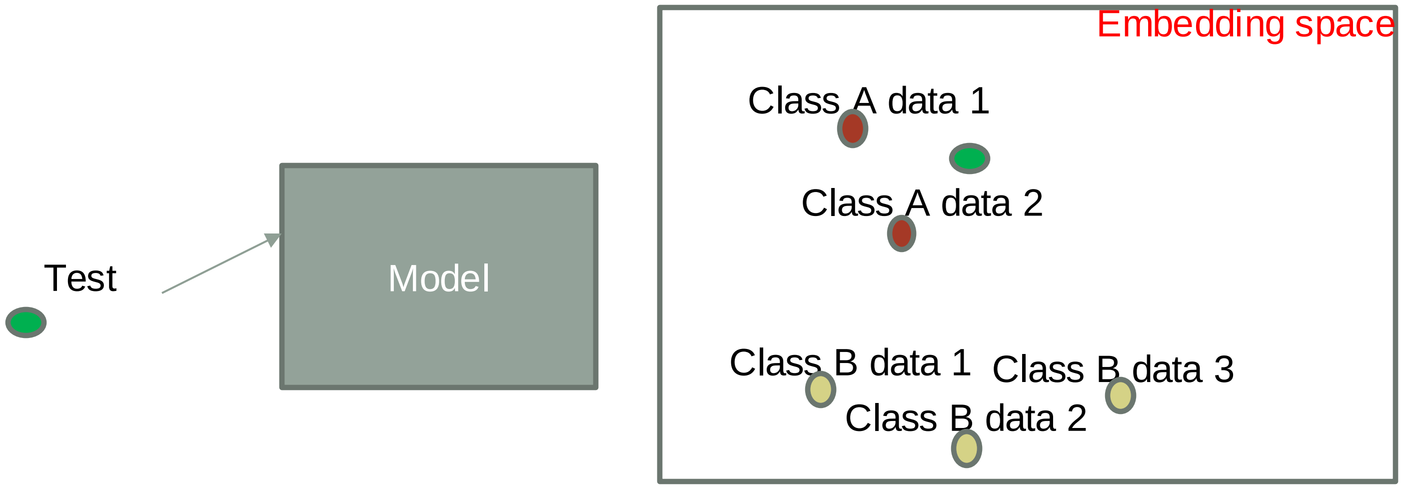
Word representation (non-contextualized)

-> Sentence representation (contextualized)

Zero/few shot classification

- With good sentence/document representations one can use it to perform zero or few shot classification





Classification benchmarks

- https://github.com/mrpeerat/Thai-Sentence-Vector-Benchmark

Thai semantic textual similarity benchmark

- We use STS-B translated ver. in which we translate STS-B from SentEval by using google-translate API
- How to evaluate sentence representation: Easy_Evaluation.ipynb
- How to evaluate sentence representation on Google Colab: https://colab.research.google.com/github/mrpeerat/Thai-Sentence-Vector-Benchmark/blob/main/SentEval.ipynb

Base Model	Spearman's Correlation (*100)	Supervised?	Latency(ms)
simcse-model-distil-m-bert	44.27		7.22 ± 0.53
simcse-model-m-bert-thai-cased	43.95		11.66 ± 0.72
simcse-model-XLMR	63.98		10.95 ± 0.41
simcse-model-wangchanberta	60.95		10.54 ± 0.33
simcse-model-phayathaibert	68.28		11.4 ± 1.01
SCT-model-XLMR	68.90		10.52 ± 0.46
SCT-model-wangchanberta	71.35		10.61 ± 0.62
SCT-model-phayathaibert	74.06		10.64 ± 0.72
SCT-Distil-model-XLMR	78.78		10.69 ± 0.48
SCT-Distil-model-wangchanberta	77.77		10.86 ± 0.55
SCT-Distil-model-phayathaibert	77.89		11.01 ± 0.62

Classification benchmarks

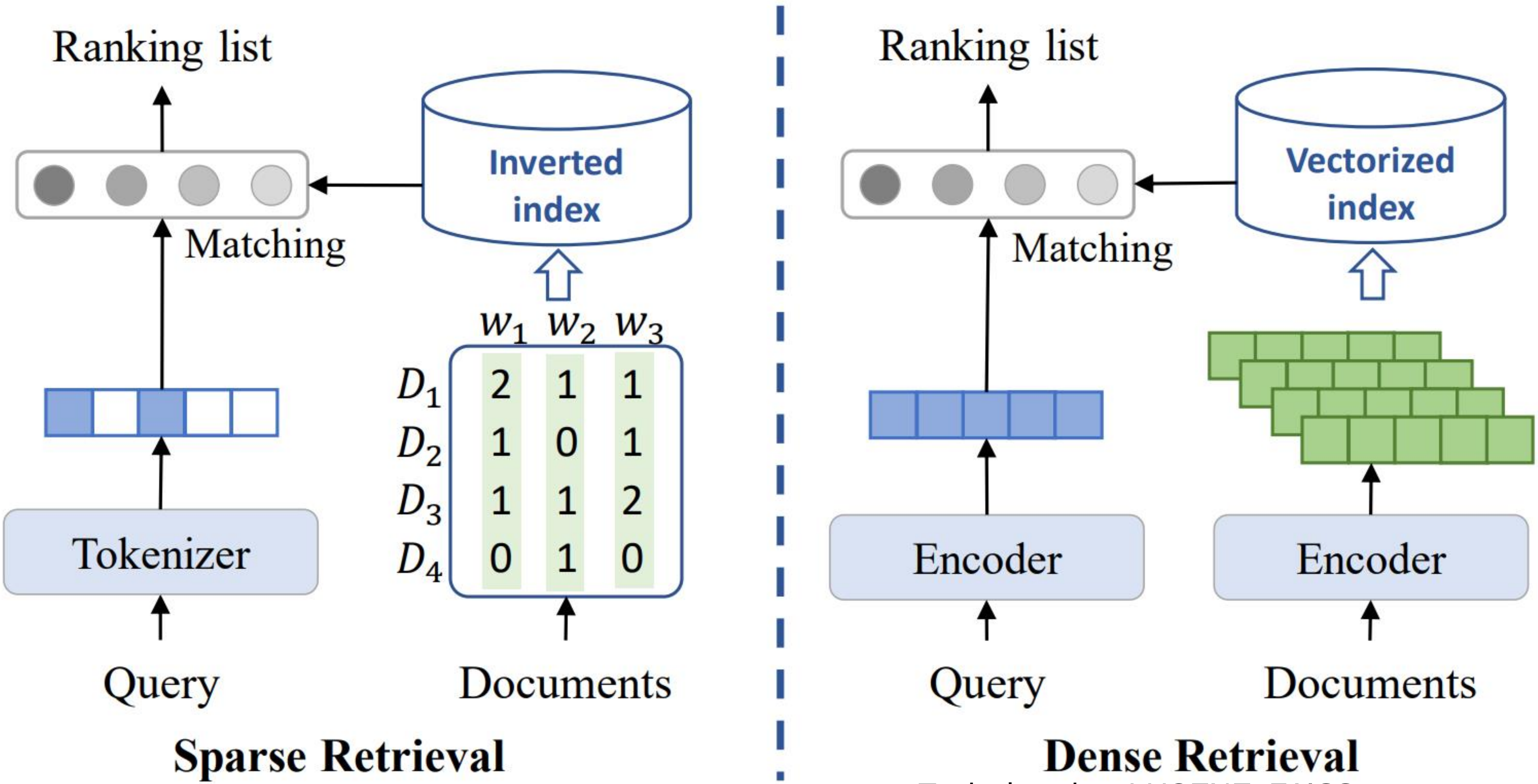
- https://github.com/mrpeerat/Thai-Sentence-Vector-Benchmark

Wongnai

Base Model	Acc (*100)	F1 (*100, weighted)	Supervised?
simcse-model-distil-m-bert	34.31	35.81	
simcse-model-m-bert-thai-cased	37.55	38.29	
simcse-model-XLMR	40.46	38.06	
simcse-model-wangchanberta	40.95	37.58	
simcse-model-phayathaibert	37.53	38.45	
SCT-model-XLMR	42.88	44.75	
SCT-model-wangchanberta	47.90	47.23	
SCT-model-phayathaibert	54.73	49.48	
SCT-Distil-model-XLMR	46.16	47.02	
SCT-Distil-model-wangchanberta	48.61	44.89	
SCT-Distil-model-phayathaibert	48.86	48.14	
SCT-Distil-model-phayathaibert-bge-m3	45.95	47.29	
ConGen-model-XLMR	44.95	46.57	
ConGen-model-wangchanberta	46.72	48.04	
ConGen-model-phayathaibert	45.99	47.54	

Search/retrieval benchmarks

Types of search



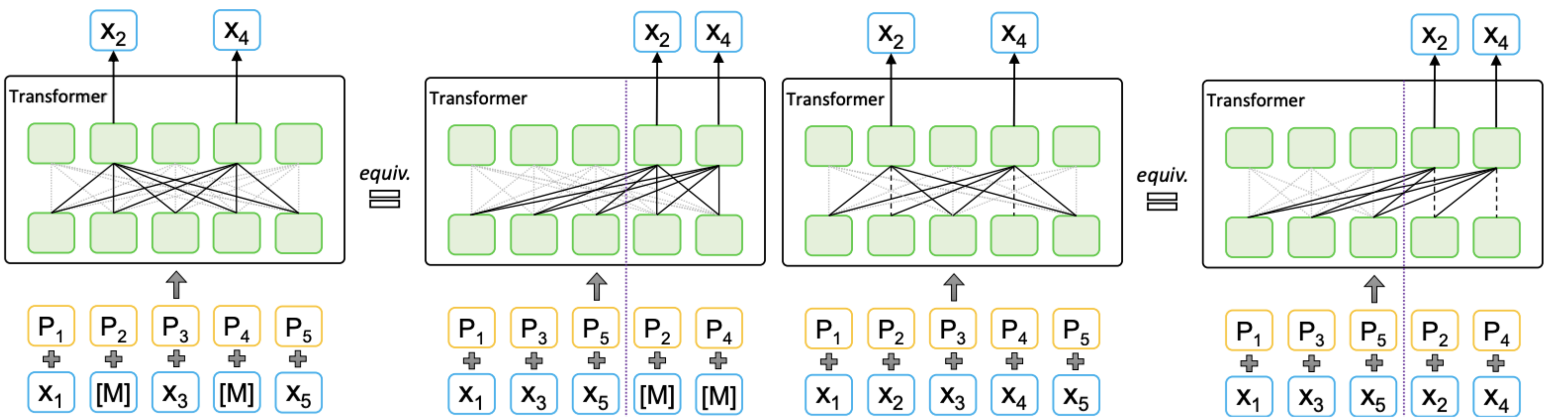
Typical engine: LUCENE

Typical engine: LUCENE, FAISS

https://github.com/facebookresearch/faiss https://arxiv.org/pdf/2004.09297

MPNET

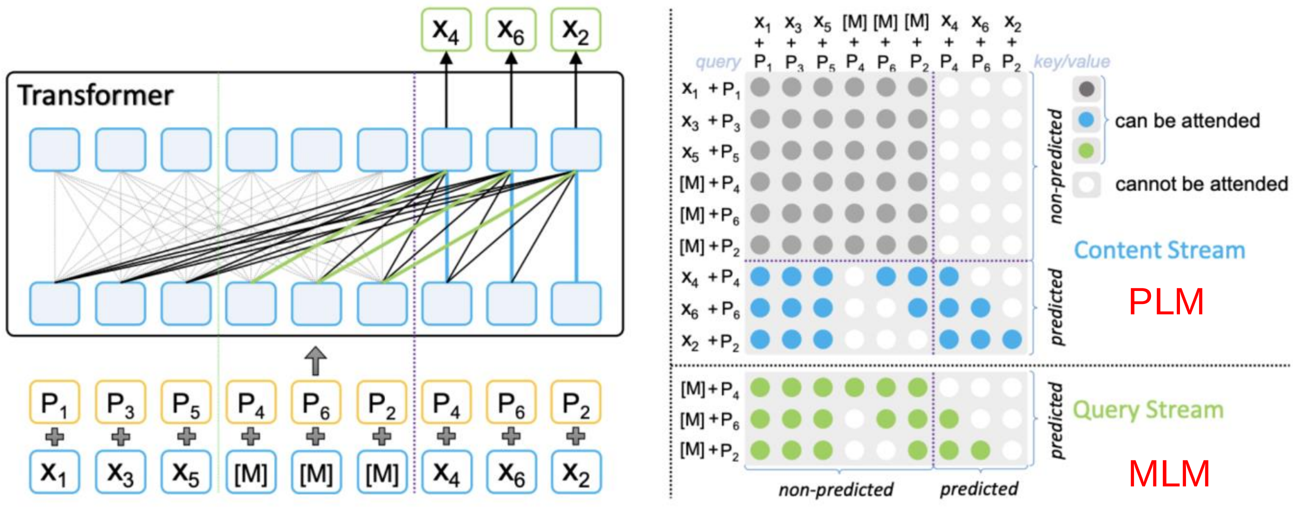
- A pretrained transformer model
  - Pretrained using Masked Langauge Modeling (MLM) and Permuted Language Modeling (PLM)
    - PLM is similar to decoder only but trained on permuted versions of the sentences.



(a) MLM (b) PLM

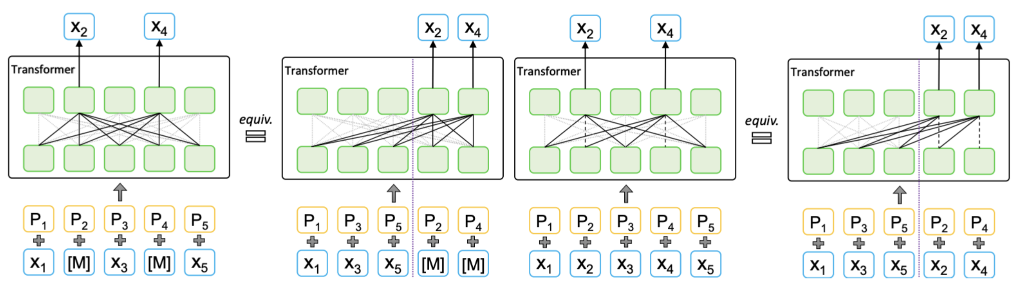
Figure 1: A unified view of MLM and PLM, where x_i and p_i represent token and position embeddings. The left side in both MLM (a) and PLM (b) are in original order, while the right side in both MLM (a) and PLM (b) are in permuted order and are regarded as the unified view.

MPNET



(a) (b)

Figure 2: (a) The structure of MPNet. (b) The attention mask of MPNet. The light grey lines in (a) represent the bidirectional self-attention in the non-predicted part $(x_{z_{<=c}}, M_{z_{>c}}) = (x_1, x_5, x_3, [M], [M], [M])$, which correspond to the light grey attention mask in (b). The blue and green mask in (b) represent the attention mask in content and query streams in two-stream self-attention, which correspond to the blue, green and black lines in (a). Since some attention masks in content and query stream are overlapped, we use black lines to denote them in (a). Each row in (b) represents the attention mask for a query position and each column represents a key/value position. The predicted part $x_{z_{>c}} = (x_4, x_6, x_2)$ is predicted by the query stream.



(a) MLM (b) PLM

Figure 1: A unified view of MLM and PLM, where x_i and p_i represent token and position embeddings. The left side in both MLM (a) and PLM (b) are in original order, while the right side in both MLM (a) and PLM (b) are in permuted order and are regarded as the unified view.

The Sentence Transformer authors then finetune MPNET using paraphrase datasets to do additional contrastive learning.

Classification Benchmarks

(delisted from pythaiNLP due to license concerns)
Chula still has Educational/research license

truevoice-intent: destination

We benchmark truevoice-intent by using `destination` as target and construct a 7-class multi-class classification. The performance is measured by micro-averaged and macro-averaged accuracy and F1 score. Codes can be run to confirm performance at this notebook. We also provide performance metrics by class in the notebook.

model	macro-accuracy	micro-accuracy	macro-F1	micro-F1
LinearSVC	**0.957806**	**0.95747712**	**0.869411**	**0.85116993**
ULMFit	0.955066	0.84273111	0.852149	0.84273111
BERT	0.8921	0.85	0.87	0.85
USE	0.943559	0.94355855	0.787686	0.802455

https://github.com/PyThaiNLP/classification-benchmarks

ConGEN: Unsupervised Control and Generalization Distillation For Sentence Representation

- Want a smaller model for sentence representation
  - But training a small model is hard

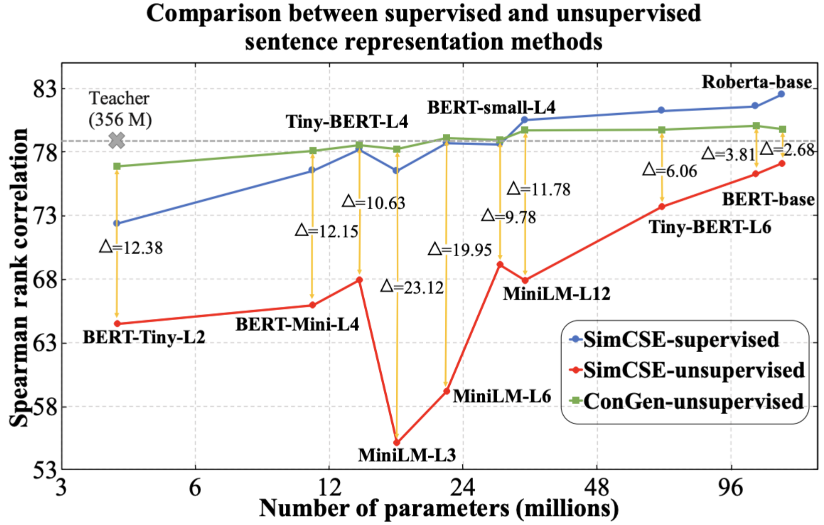
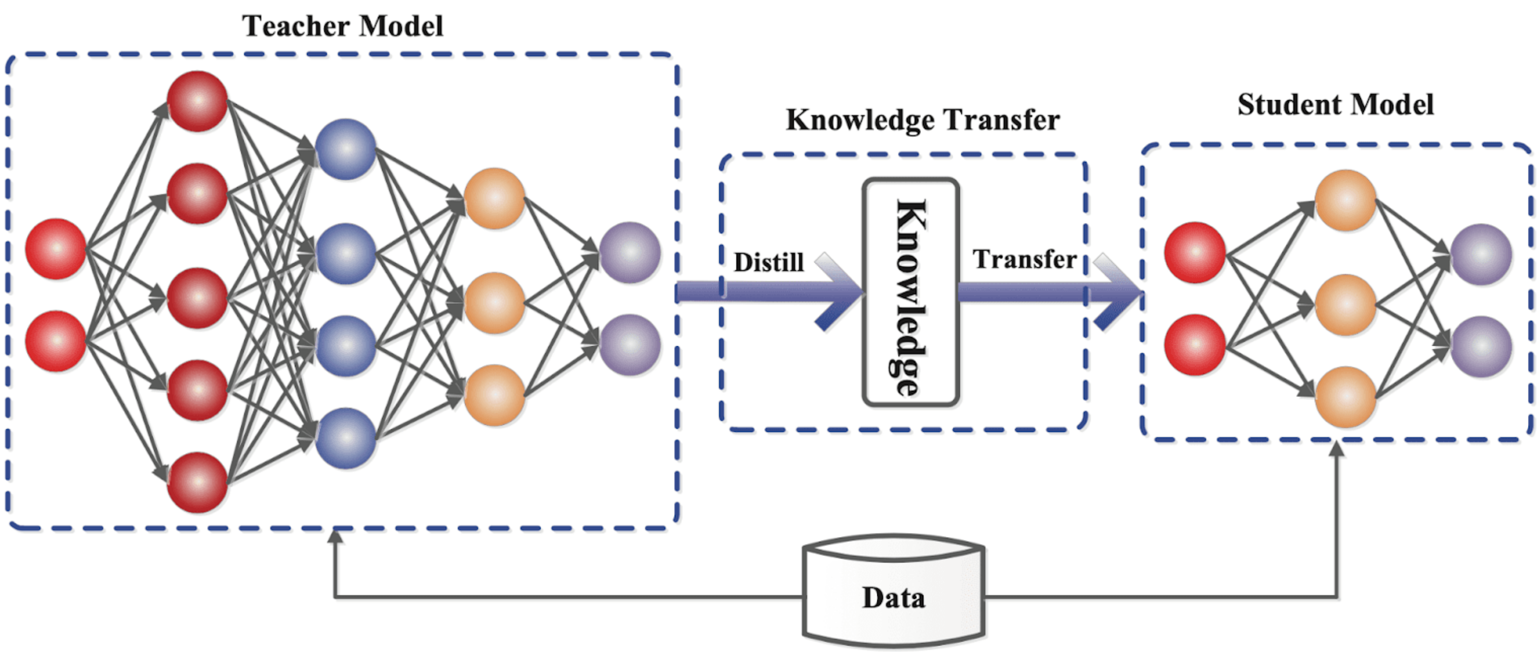


Figure 1: Comparison between finetuning LMs (SimCSE) vs. knowledge distillation (ConGen) on the average of 7 semantic textual similarity (STS) benchmark datasets and Δ is the improvement of ConGen from SimCSE.

https://aclanthology.org/2022.findings-emnlp.483/

Knowledge Distillation

- Student model learns from a teacher model
  - Training a small model is hard, but training a larger sophisticate model is easy.
  - Have the teacher teach the smaller model.



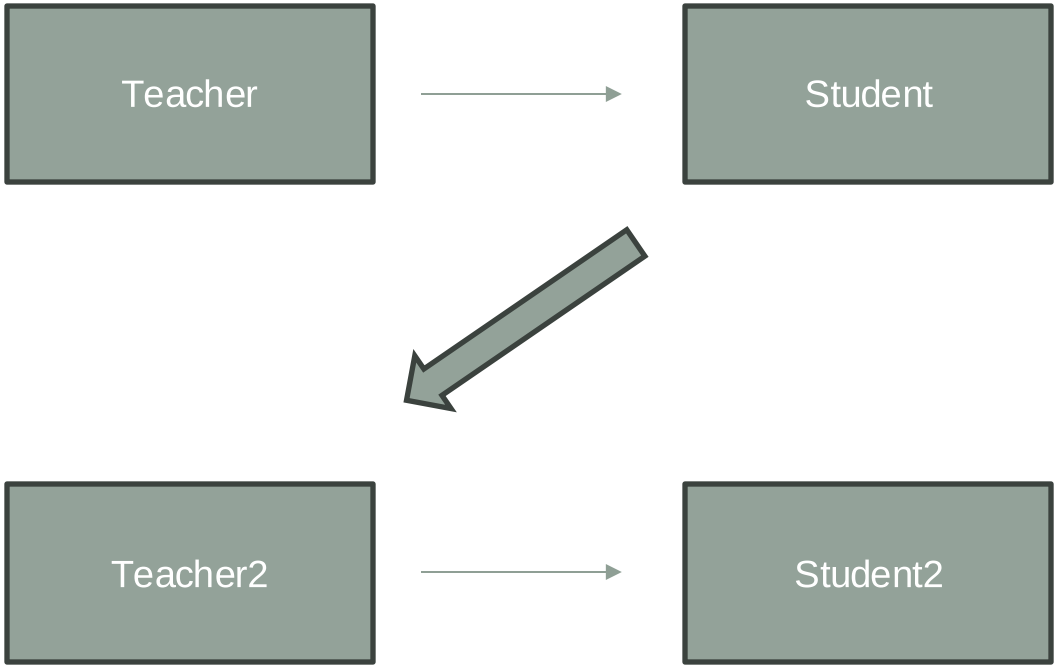
https://blog.roboflow.com/what-is-knowledge-distillation/

Distillation use cases

- Distillation for smaller model

Distillation use cases

- Distillation for model improvement (self-distillation)
  - Keep re-initializing the teacher model



Instance Queue

- In contrastive learning, a large mini-batch is preferred.
  - Every mini-batch new samples need to be computed <- compute bounded.
- We can keep some of the old embeddings in a queue

ConGen

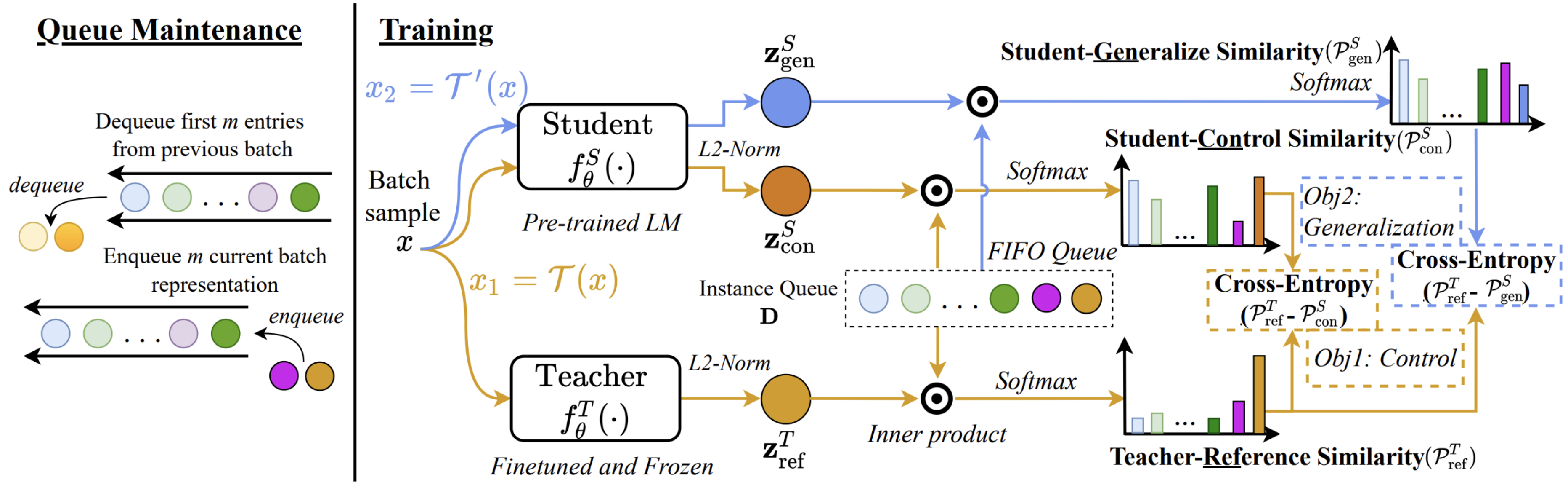


Figure 2: Illustration of *Control and Generalization Distillation (ConGen)* training pipeline. For the teacher model, we freeze the weights during the distillation. We train student model by minimizing the cross-entropy of teacher & student similarity distributions computed over an instance queue.

Backtranslate is a very good augmentation method

Model	STS average scores	
	BERT-Tiny	**BERT-base**
Baseline		
EN→DE→EN (Google NMT)	**76.85**	**80.06**
Other augmentation methods		
EN→DE→EN (MBart)	71.35	75.37
MLM 15%	74.99	78.44
Synonym replacement	76.01	80.01
Crop 10%	76.14	79.95
Word deletion 10%	76.15	**80.06**
Delete one word	76.14	80.02

Table 6: Comparison between data augmentation operations for the generalize objective.

Outline

- Naïve Bayes
- Neural methods
- Topic Models
  - Latent topic models (LDA)

Text classification and language modeling

- P(x|c)

x_1 is a feature that looks at the first word

- P(x = ยอด| c=5) = $\dfrac{\text{count(x = ยอด , c = 5)}}{\text{count(c = 5)}}$

List of words

List of classes

|X| |c|

- P(c)
- P(c = 5) = $\dfrac{\text{count (c = 5)}}{\text{count (all reviews)}}$

This looks like… n-grams, but instead of conditioning on the past, we condition on the topic –
bag of words model for topic modeling (unigram with topic)

Language modeling view

- Which class is this review
- P(w|c)

อร่อย แต่ ไม่ ถูก

Class= 1	
อร่อย	0.01
แต่	0.4
ไม่	0.4
ถูก	0.03
…	

Class= 5	
อร่อย	0.4
แต่	0.05
ไม่	0.25
ถูก	0.15
…	

P(s|c =1) = 0.01*0.4*0.4*0.03 = 0.000048
P(s|c = 5) = 0.4*0.05*0.25*0.15 = 0.00075

Topic modeling

- Sometimes you want to model the topic of a document

Class= บรรยากาศ

อร่อย 0.01
แต่ 0.4
ไม่ 0.4
ถูก 0.03
…

Class= อาหาร

อร่อย 0.4
แต่ 0.05
ไม่ 0.25
ถูก 0.15
…

อาหารที่นี่ไม่ค่อยอร่อย แต่ขนมใช้ได้เลย ถ้าว่างอาจจะกลับมากินอีก แนะนำให้สั่งเค้กใบเตย

ด้านบรรยากาศ มีเสียงก่อสร้างมาจากตึกข้างๆ แต่นอกนั้นตกแต่งโอเค แต่ยังขาดอะไรไปหลายๆอย่าง

P(s|c=บรรยากาศ) = ?

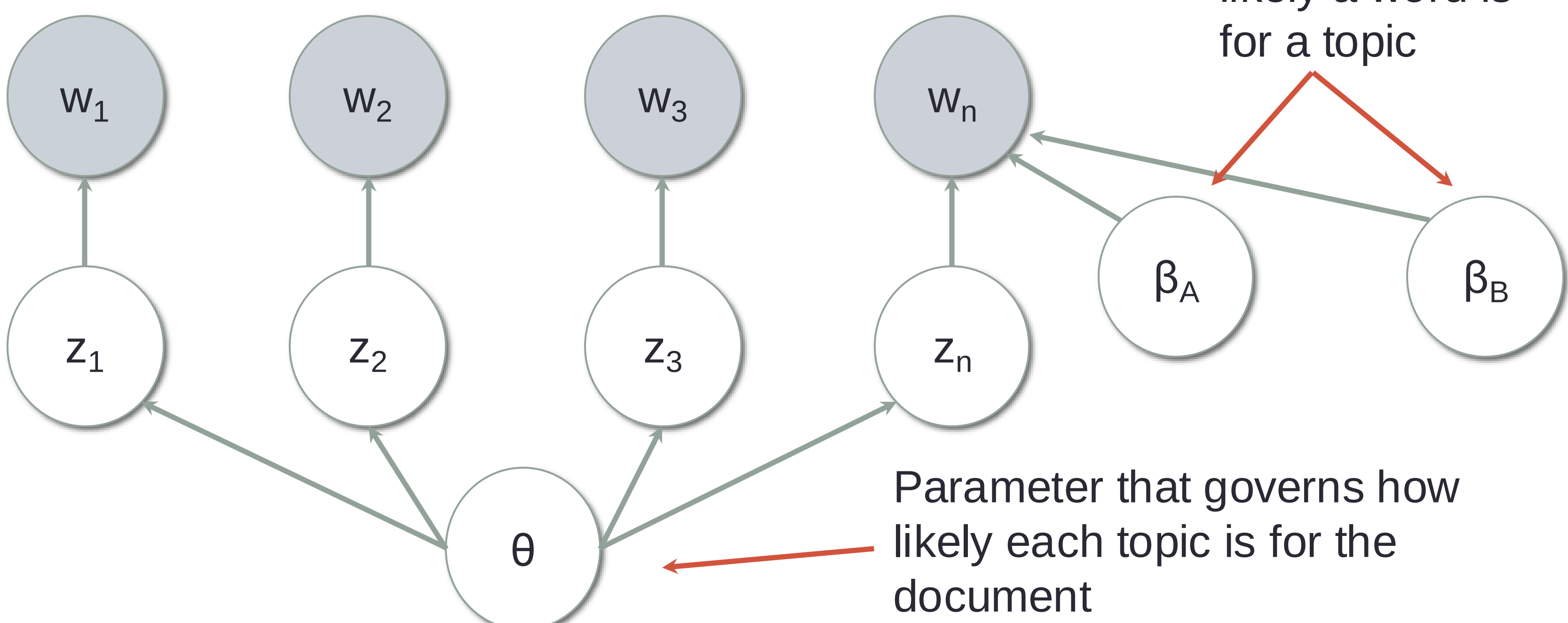
P(s|c=อาหาร) = ?

Naïve Bayes Topic modeling issues

- Most document have multiple topics (multi-label).
  - Our model assumes 1 document 1 topic.
- Solution: Let a document be a mixture of topics (language model interpolation). Each word has its own topic, z.
    - P(w) = P(w is topic A) P(w | topicA) + P(w is topic B) P(w | topicB)
    - P(w is topic A) + P(w is topic B) = 1

Naïve Bayes Topic modeling issues

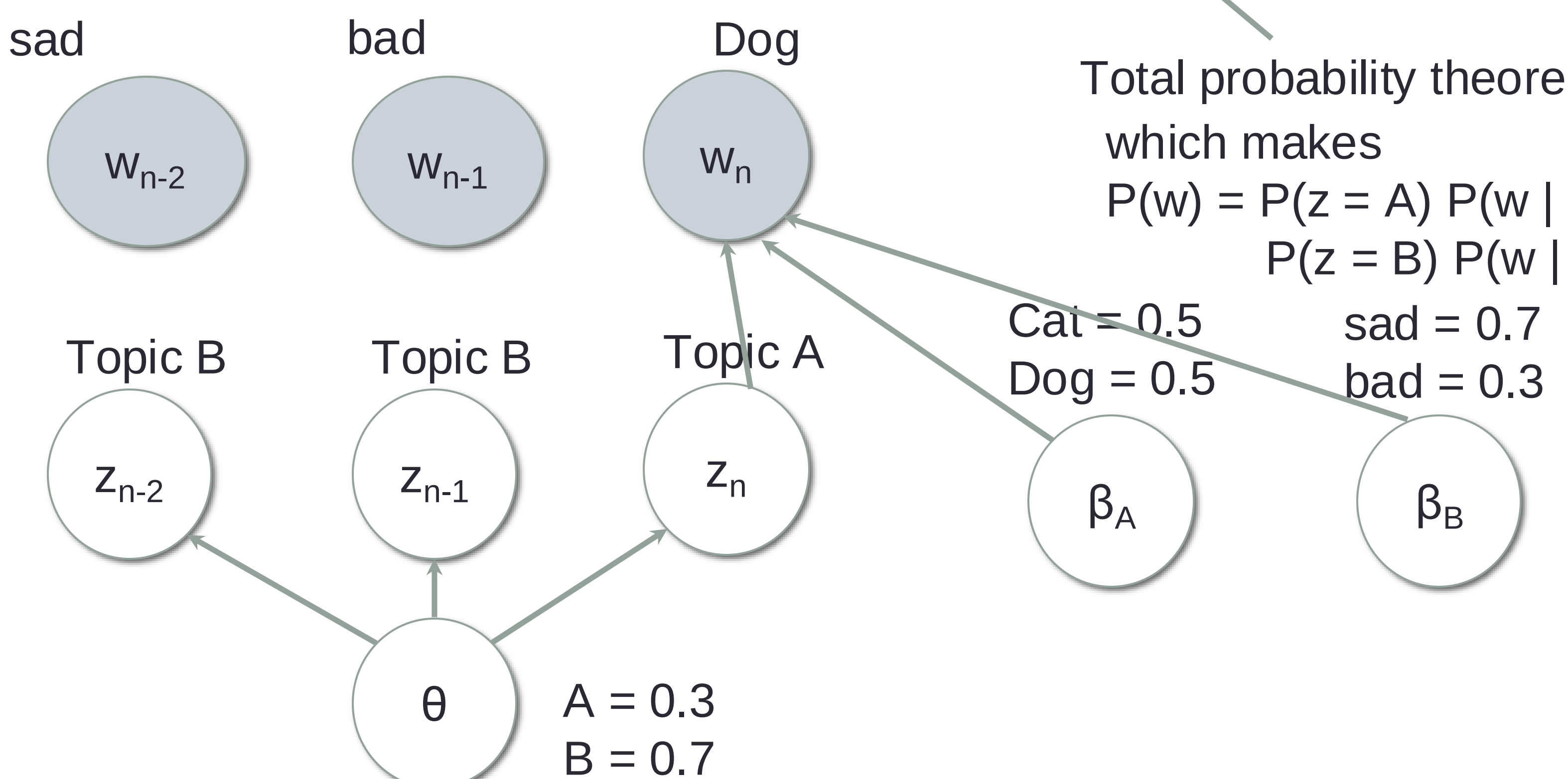
- Most document have multiple topics. Our model assumes 1 document 1 topic.
  - Let a document be a mixture of topics (language model interpolation). Each word has its own topic, z.
    - $P(w) = P(z = A)\ P(w \mid z = A) + P(z = B)\ P(w \mid z = B)$
    - $P(z = A) + P(z = B) = 1, \quad \theta = P(z = A) \quad \beta_A = P(w \mid z = A)$



Graphical model and generation

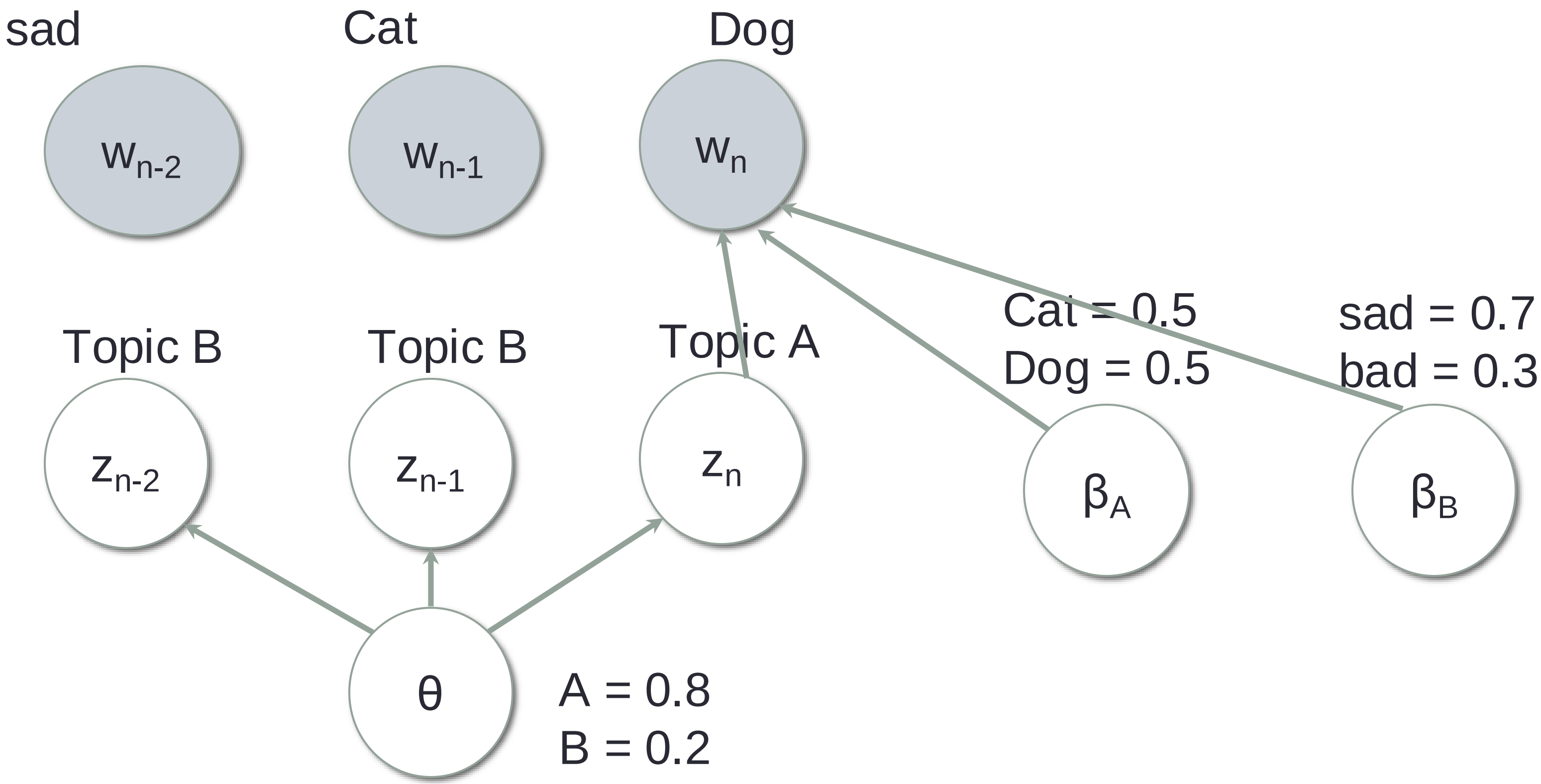
How likely a sentence is likely to be generated follows this generation process
P(sad,bad,Dog,B,B,A) = P(B)P(B)P(A)P(sad|B)P(bad|B)P(Dog|A)
Note
P(sad,bad,Dog) = P(sad,bad,Dog,A,A,A) + P(sad,bad,Dog,A,A,B)
P(sad,bad,Dog,A,B,A) + P(sad,bad,Dog,A,B,B) +...

Total probability theorem
which makes
P(w) = P(z = A) P(w | z = A) +
P(z = B) P(w | z = B)



pLSA (probabilistic Latent Semantic Analysis) model

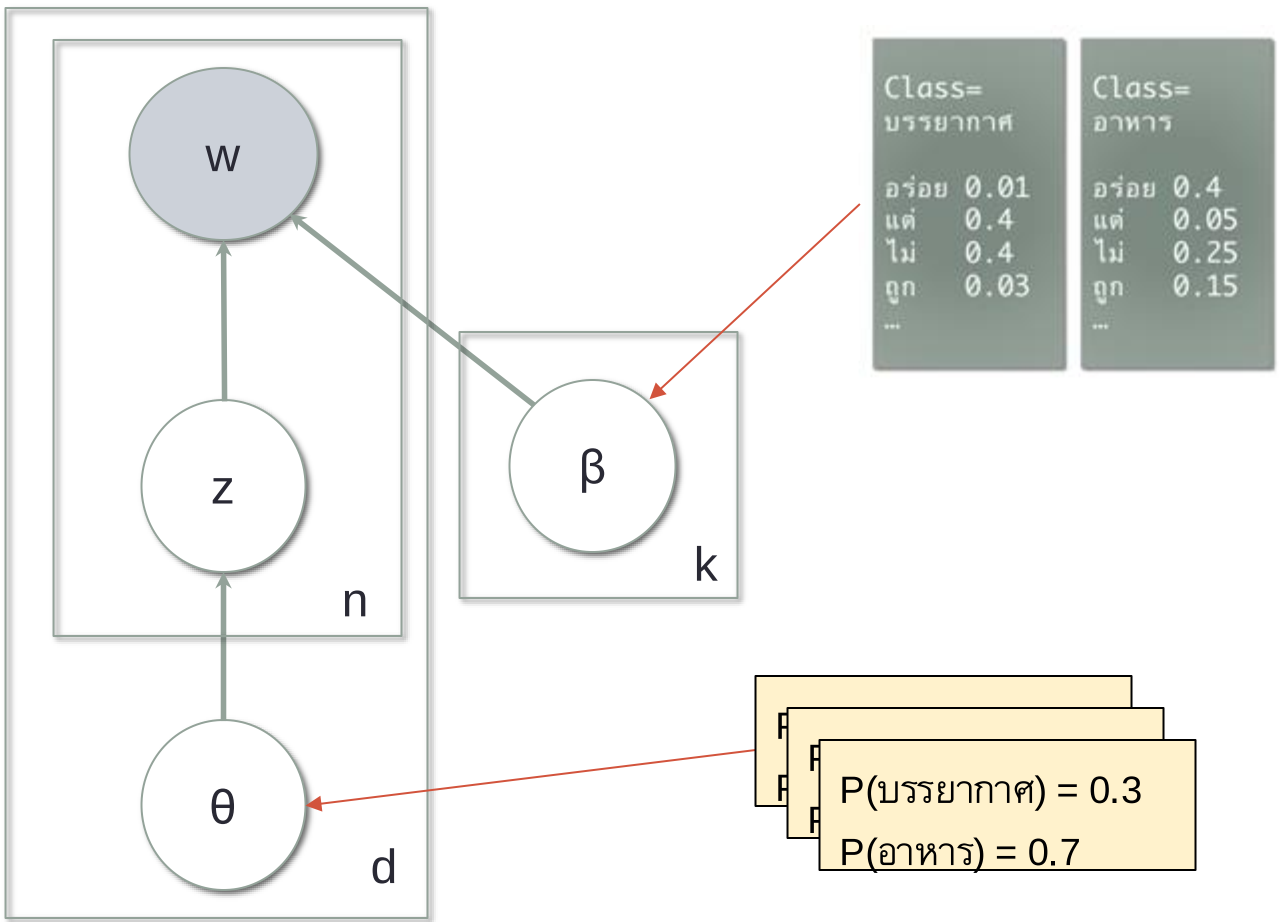
pLSA models a document with their own topic mixture θ



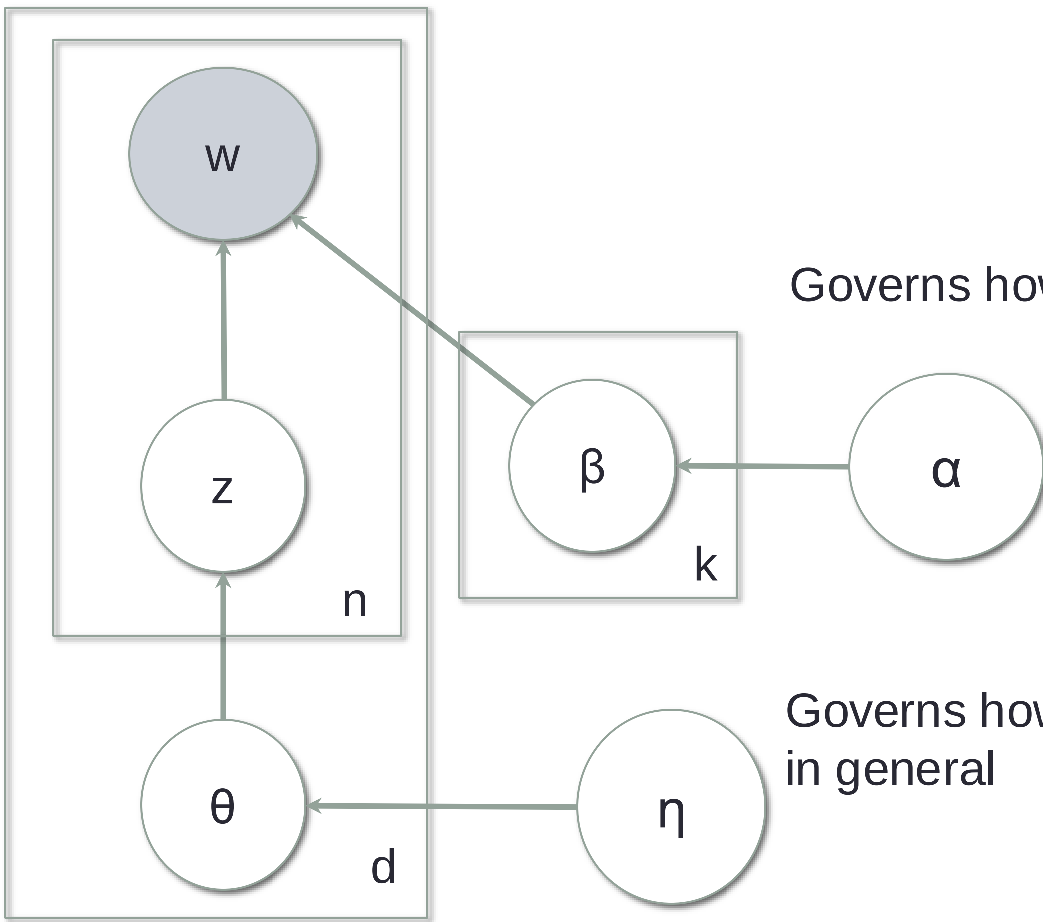
pLSA

- pLSA automatically clusters words into topic unigrams
  - Requires user to specify number of topics
- Automatically learn document representation based on the learned topics
  - DocA = [0.7 0.3] DocB = [0.2 0.8] DocC = [0.5 0.5]
- Overfits easily to data outside of the training set
  - Nothing that ties all document together
  - A document from a document collection should be have topic distributions that are similar
- Solution: LDA (Latent Dirichlet Allocation)

pLSA

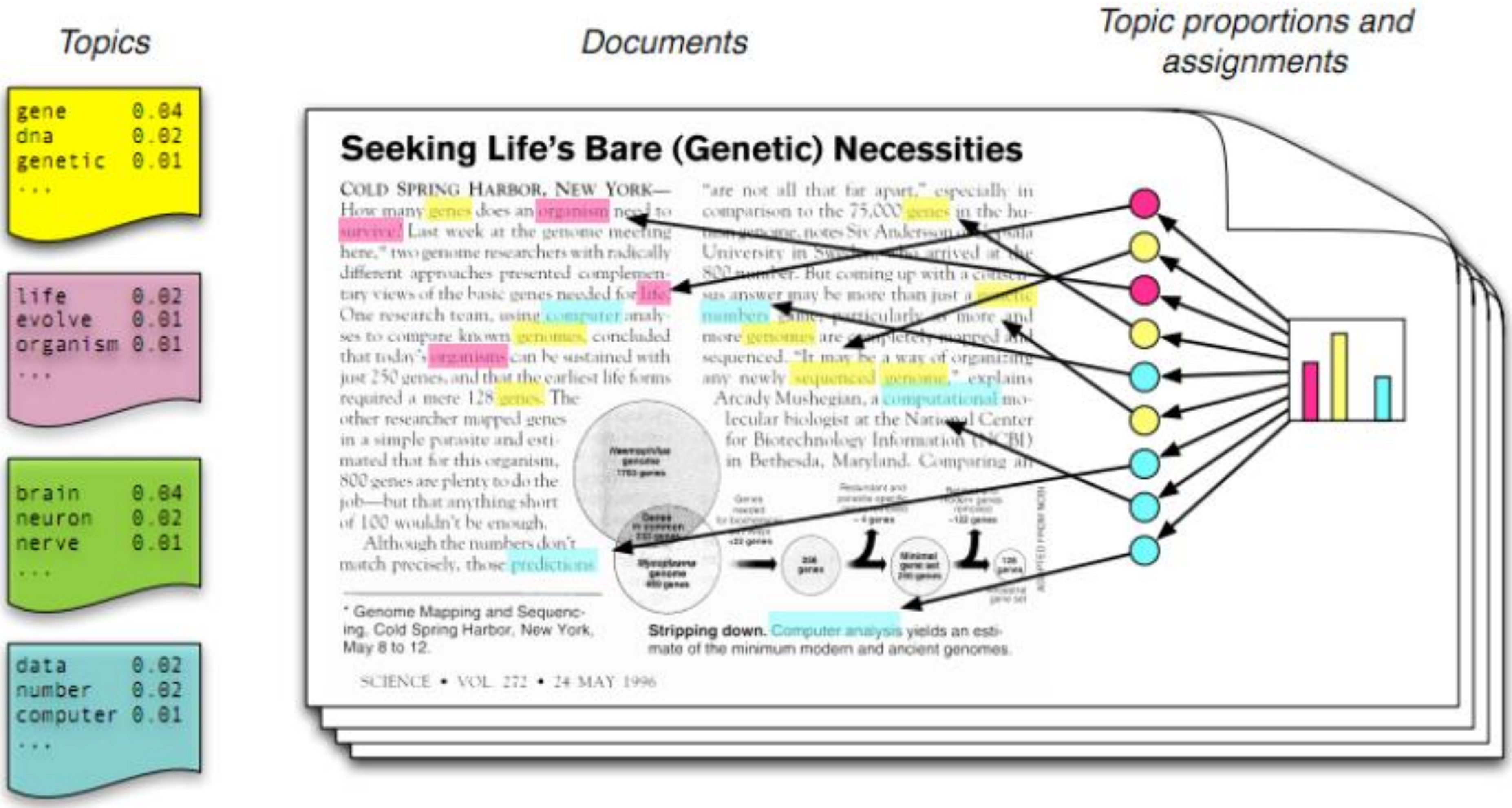

w
z
θ
n
d
β
k
Class=
บรรยากาศ
อร่อย 0.01
แต่ 0.4
ไม่ 0.4
ถูก 0.03
...
Class=
อาหาร
อร่อย 0.4
แต่ 0.05
ไม่ 0.25
ถูก 0.15
...
P(บรรยากาศ) = 0.3
P(อาหาร) = 0.7

LDA



Governs how topics should be in general

Governs how documents should be in general



Introduction to Probabilistic Topic Models, Blei 2011
http://menome.com/wp/wp-content/uploads/2014/12/Blei2011.pdf

LDA

- Automatically learns topics, and the word distribution of each topic
  - Just give a bunch of documents!
  - Each document is given a mixture of topics
    - Dirichlet prior prefers sparse topics – each document only have probability in few topics – easy for interpretability
- Requires user to pick number of topic
- Requires user to make sense of the learned topics
  - For more information on how to help visualize/evaluate unsupervised topic models
    - https://youtu.be/UkmIljRIG_M

Unsupervised topic modeling for real estate

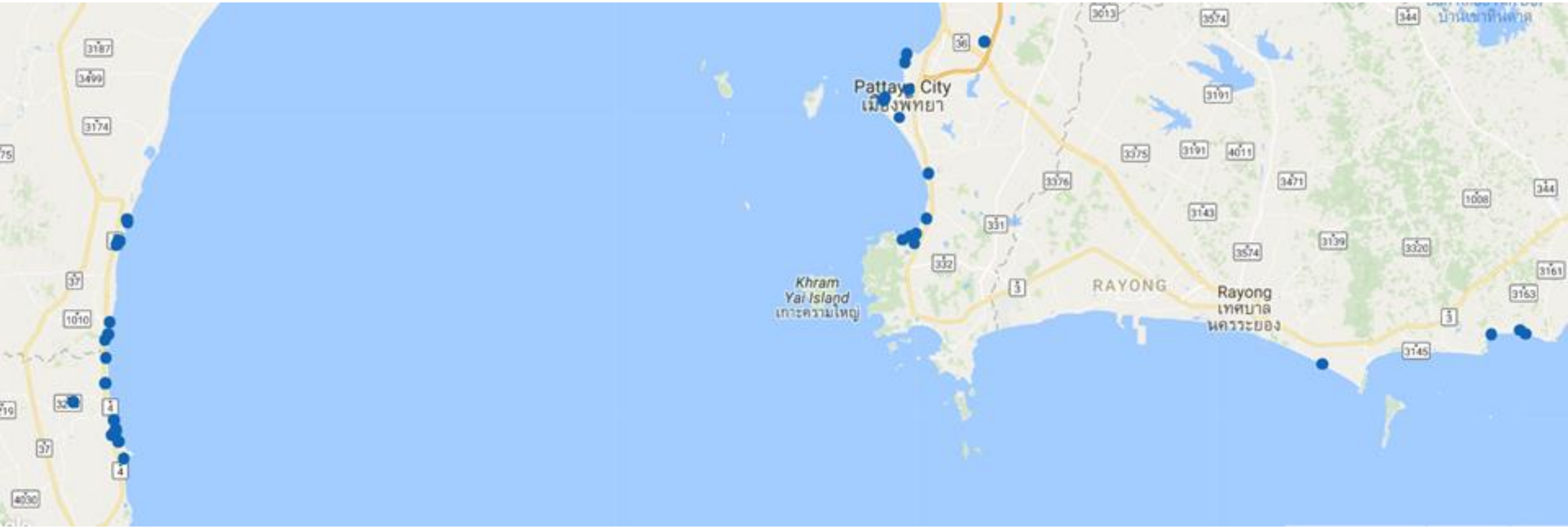
Can we learn real estate characteristics from unstructured data?

คอนโดหรูสไตล์อังกฤษ แห่งแรกในเขาใหญ่ ที่ติด ถ.ธนะรัชต์ มากที่สุด 1 ห้องนอน 1 ห้องน้ำ 1 ห้องนั่งเล่น พร้อมห้องครัวแยกเป็นสัดส่วน

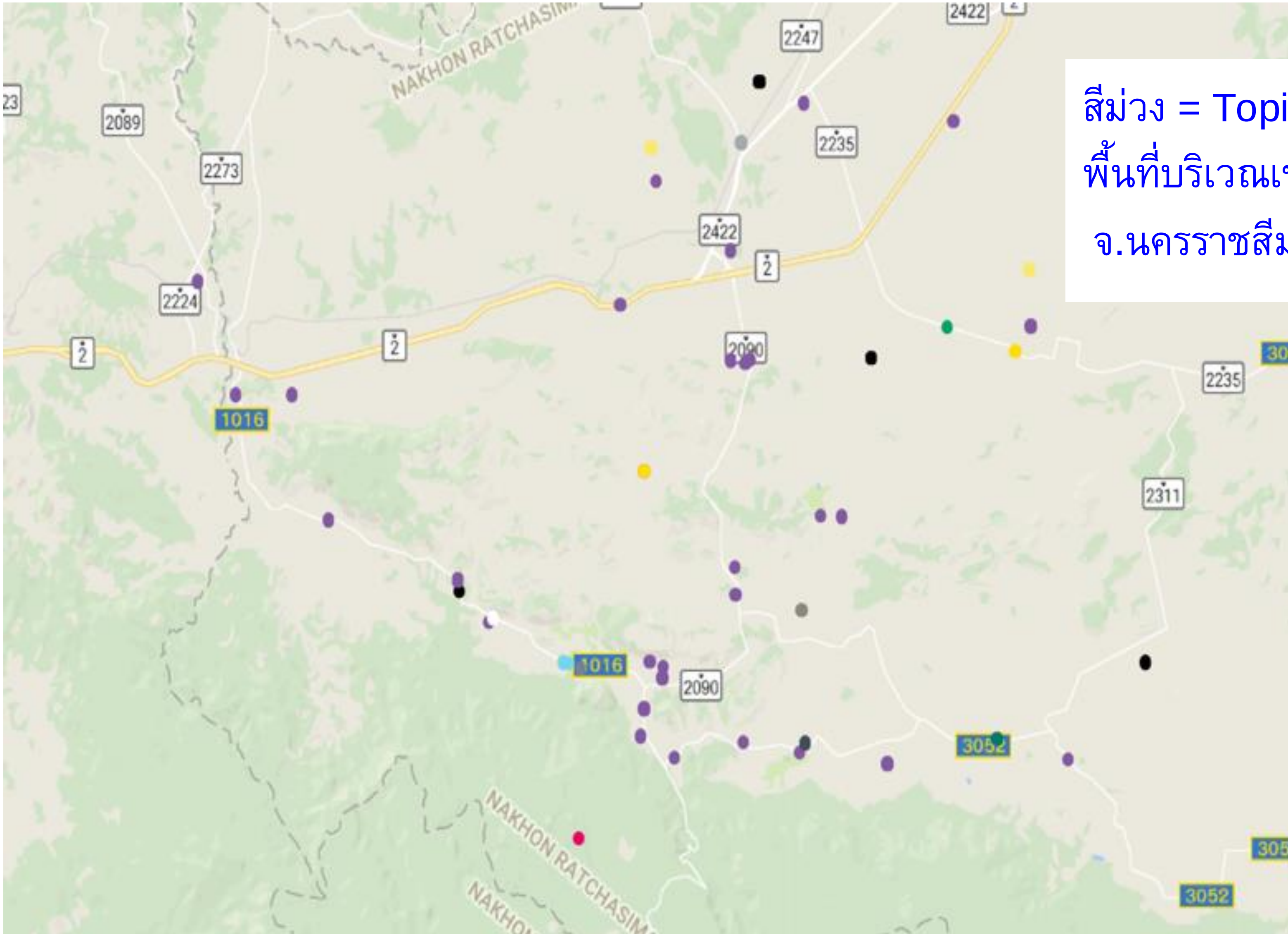
Just give it a bunch of descriptions

LDA Examples

```
Topic 28
0.068*"วิว" + 0.058*"ทะเล" + 0.038*"คอนโด" + 0.029*"หัว" + 0.027*"คอนโดมิเนียม" + 0.025*"มองเห็น" + 0.023*"ทัศนียภาพ" + 0.
022*"ชายหาด"
```



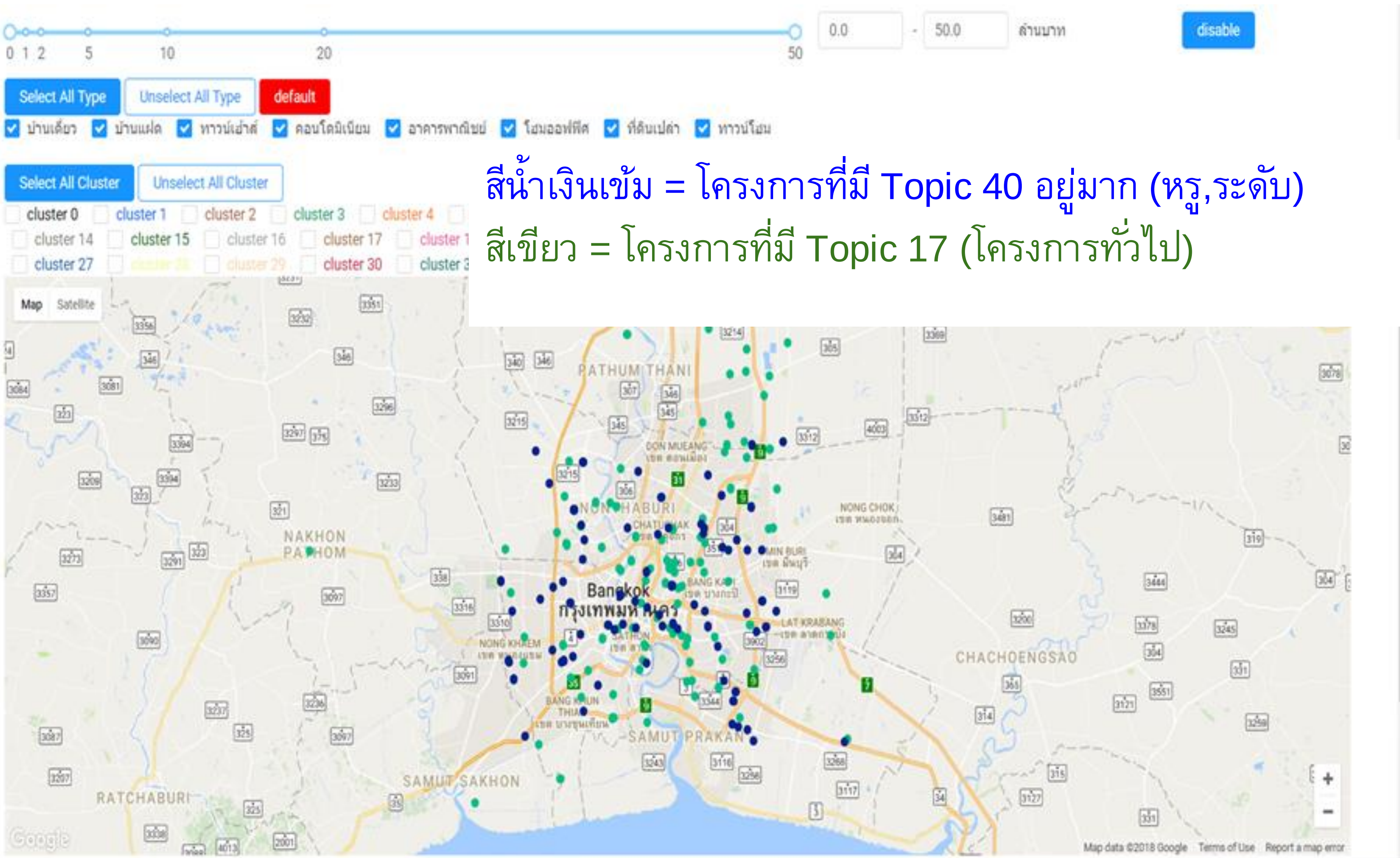
```
Topic 9
0.071*"ธรรมชาติ" + 0.031*"บรรยากาศ" + 0.028*"ร่มรื่น" + 0.027*"บ้าน" + 0.025*"ท่ามกลาง" + 0.025*"สวน" + 0.025*"สัมผัส" + 0.
021*"พื้นที่"
```



สีม่วง = Topic 9
พื้นที่บริเวณเขาใหญ่
จ.นครราชสีมา

Topic 40
0.115*"ระดับ" + 0.066*"เหนือ" + 0.046*"หรู" + 0.031*"ทำเล" + 0.026*"ชีวิต" + 0.026*"ใช้ชีวิต" + 0.016*"สไตล์" + 0.016*"สะท้อน"

Topic 17
0.077*"พื้นที่" + 0.060*"ออกแบบ" + 0.045*"โล่ง" + 0.039*"โปร่ง" + 0.038*"ใช้สอย" + 0.020*"ประโยชน์" + 0.018*"ห้อง" + 0.017*"อาคาร"



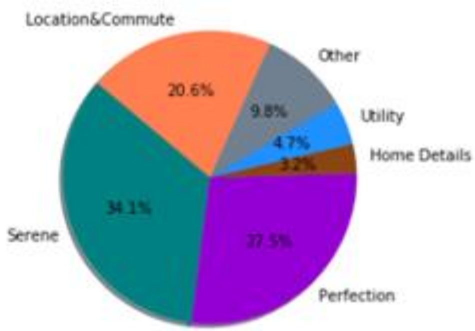
สีน้ำเงินเข้ม = โครงการที่มี Topic 40 อยู่มาก (หรู,ระดับ)

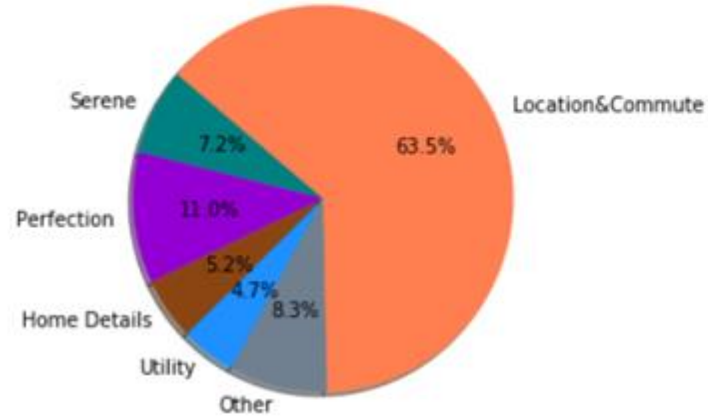
สีเขียว = โครงการที่มี Topic 17 (โครงการทั่วไป)

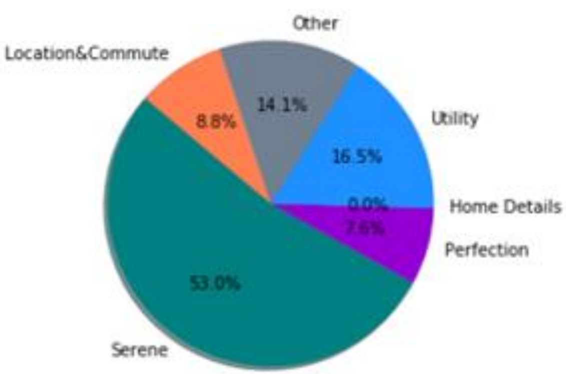
Time and advertising trends

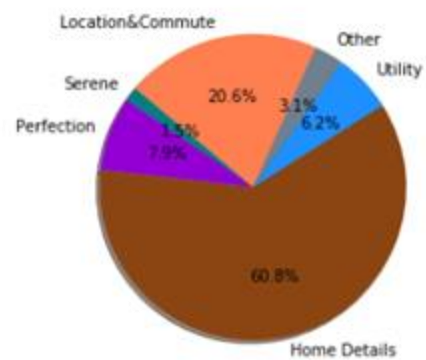
Advertisement niche of each developer

Each real estate developer has its own style









More ideas

Uniting the Tribes: Using Text for Marketing Insight

Jonah Berger, Ashlee Humphreys, Stephan Ludwig, more...

Show all authors

First Published August 29, 2019 | Research Article | Check for updates

https://doi.org/10.1177/0022242919873106

Article information

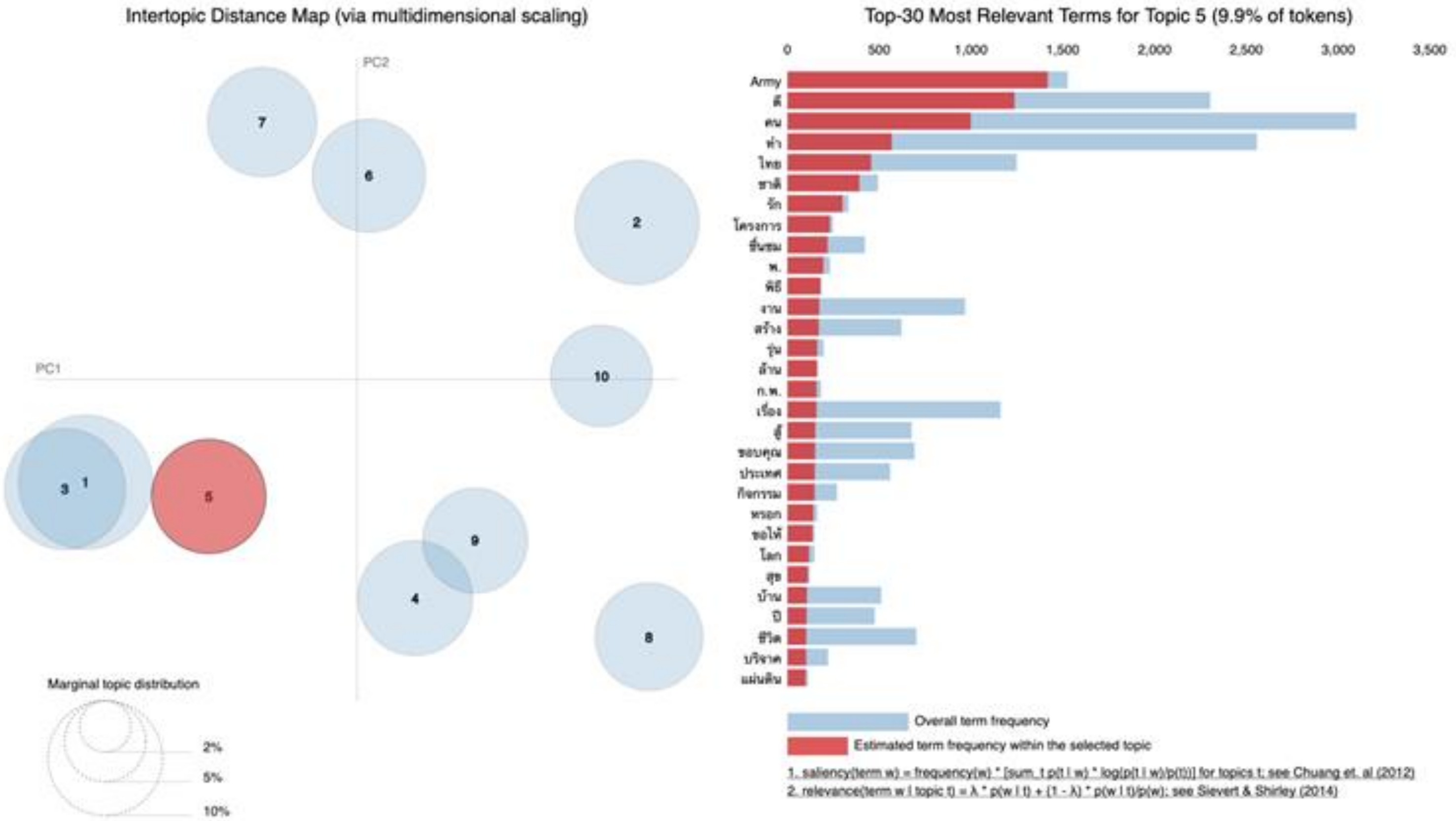
Abstract

Words are part of almost every marketplace interaction. Online reviews, customer service calls, press releases, marketing communications, and other interactions create a wealth of textual data. But how can marketers best use such data? This article provides an overview of automated textual analysis and details how it can be used to generate marketing insights. The authors discuss how text reflects qualities of the text producer (and the context in which the text was produced) and impacts the audience or text recipient. Next, they discuss how text can be a powerful tool both for prediction and for understanding (i.e., insights). Then, the authors overview methodologies and metrics used in text analysis, providing a set of guidelines and procedures. Finally, they further highlight some common metrics and challenges and discuss how researchers can address issues of internal and external validity. They conclude with a discussion of potential areas for future work. Along the way, the authors note how textual analysis can unite the tribes of marketing. While most marketing problems are interdisciplinary, the field is often fragmented. By involving skills and ideas from each of the subareas of marketing, text analysis has the potential to help unite the field with a common set of tools and approaches.

https://journals.sagepub.com/doi/full/10.1177/0022242919873106

https://stacks.stanford.edu/file/druid:ym245nv3149/twitter-TH-202009.pdf

Tweet analysis



Visualized using pyLDAvis https://github.com/bmabey/pyLDAvis

https://www.facebook.com/teattapol/posts/3337686199651109?_rdc=1&_rdr

LDA with deep learning

- LDA was develop on discrete inputs (words)
  - Modified to work with dense representation (word vectors)
    - “Gaussian LDA for Topic Models with Word Embeddings”
    - http://www.aclweb.org/anthology/P15-1077

- Modified network structure and loss function to include LDA traits
  - LDA2vec https://multithreaded.stitchfix.com/blog/2016/05/27/lda2vec/

Clustering on document embeddings?

https://aclanthology.org/2024.findings-emnlp.790.pdf
https://github.com/ddangelov/Top2Vec
https://arxiv.org/pdf/2008.09470

- Top2Vec proposes a simple method to cluster document embeddings
  - Use UMAP+HDBSCAN to identify number of clusters and the cluster
  - Represents cluster using most representative word in the cluster

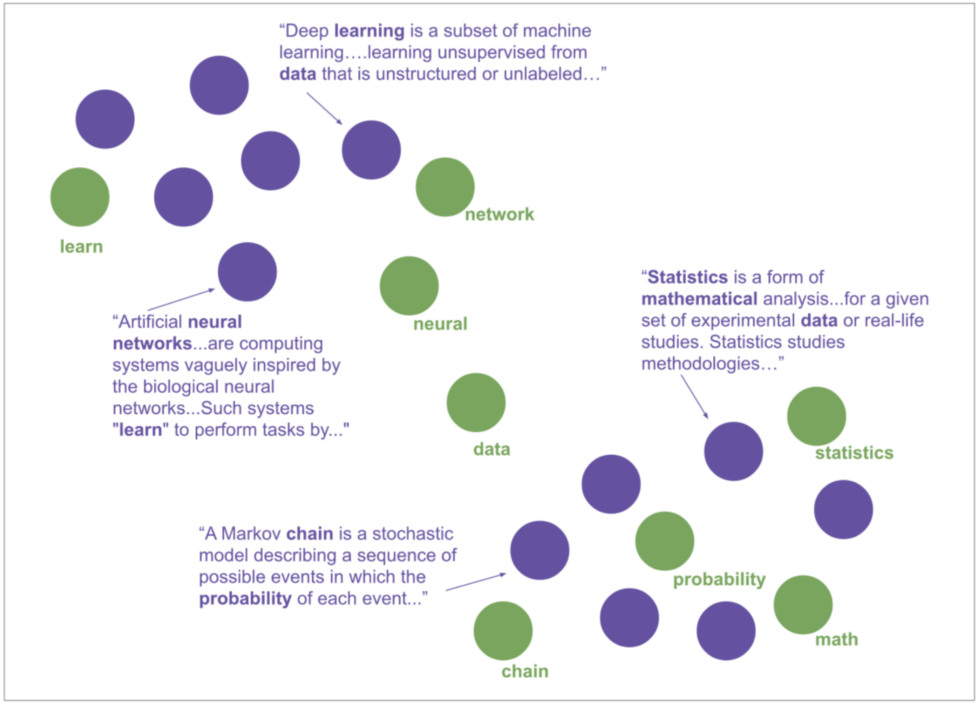


Figure 1: An example of a semantic space. The purple points are documents and the green points are words. Words are closest to documents they best represent and similar documents are close together.

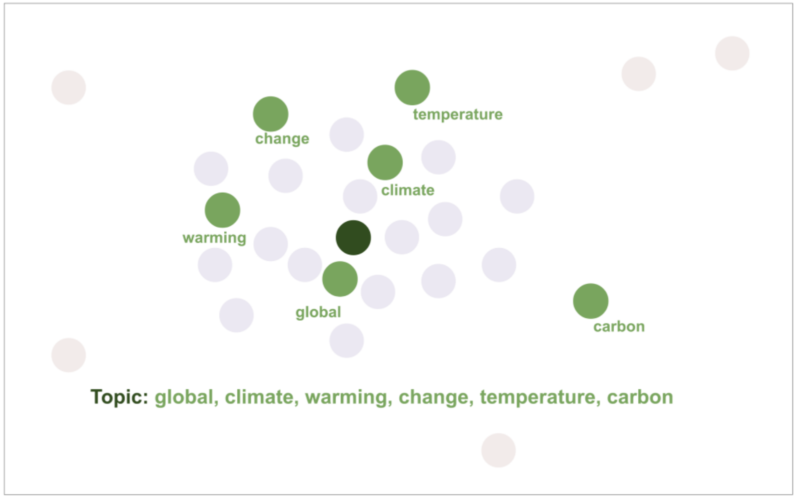
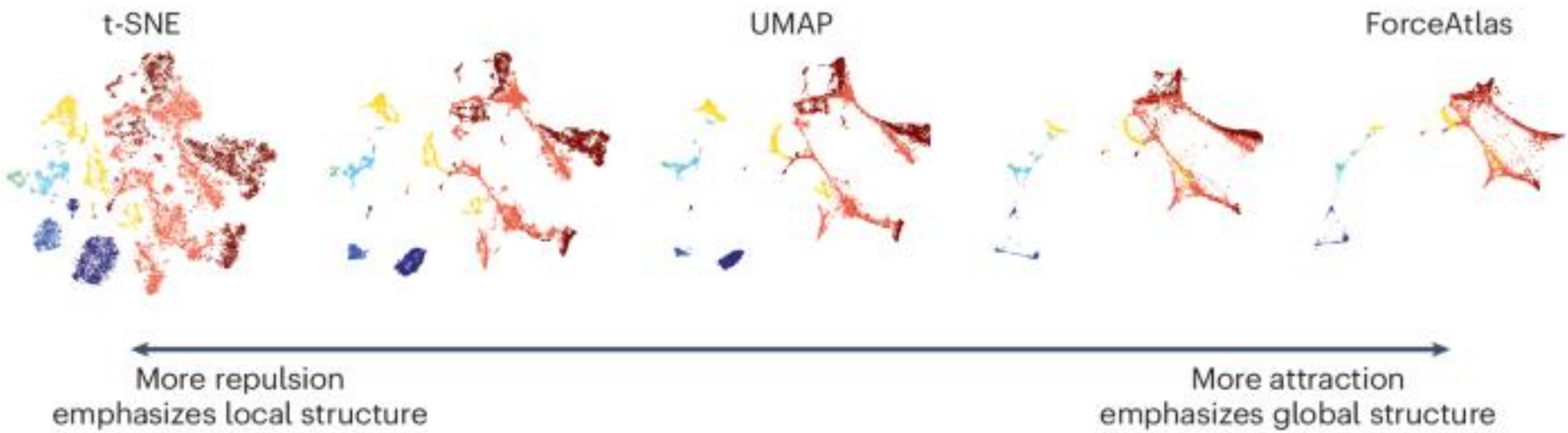


Figure 5: The topic words are the nearest word vectors to the topic vector.

UMAP

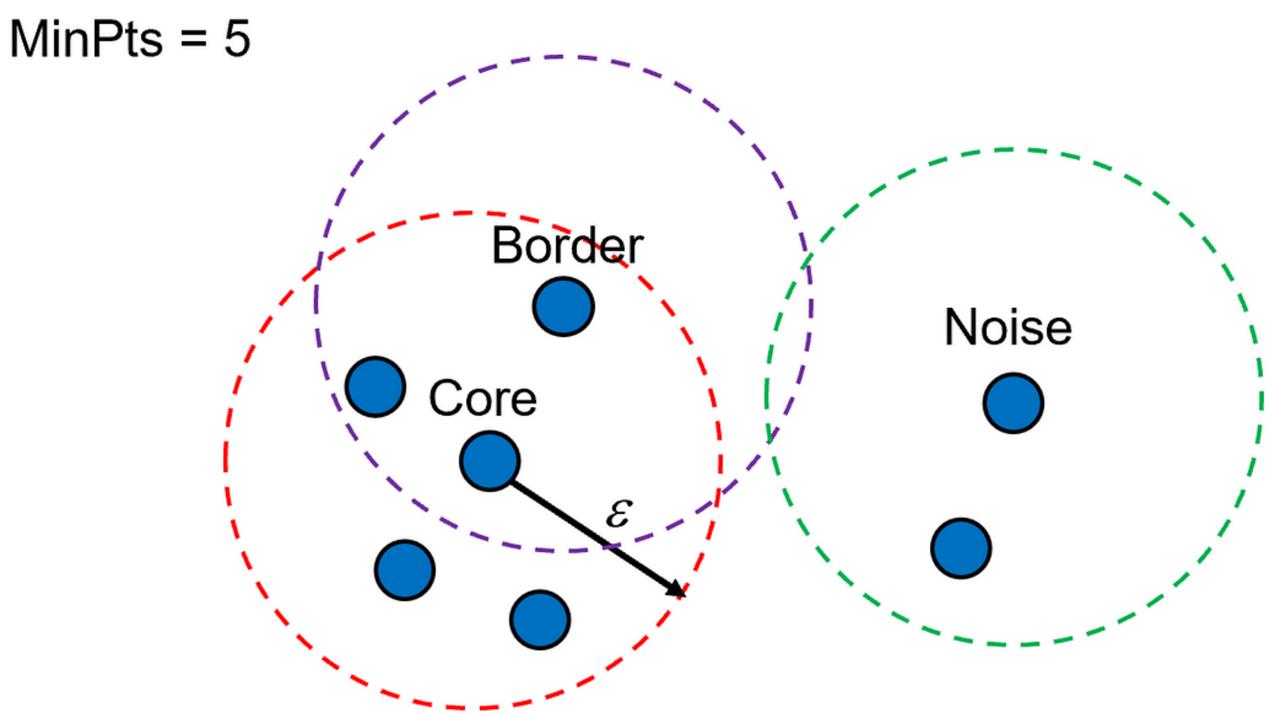
- UMAP is a data visualization/dimensionality reduction technique that focuses on preserving local and global structure of high dimensional data



https://www.nature.com/articles/s41592-024-02301-x

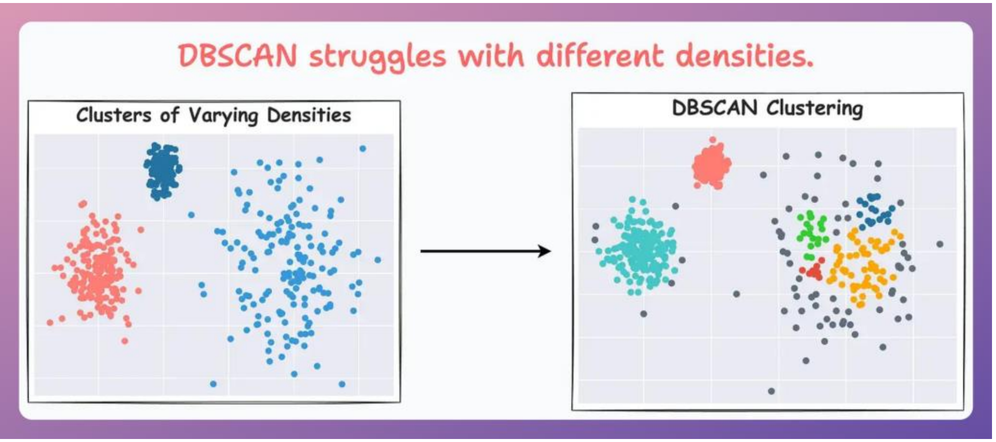
HDBSCAN

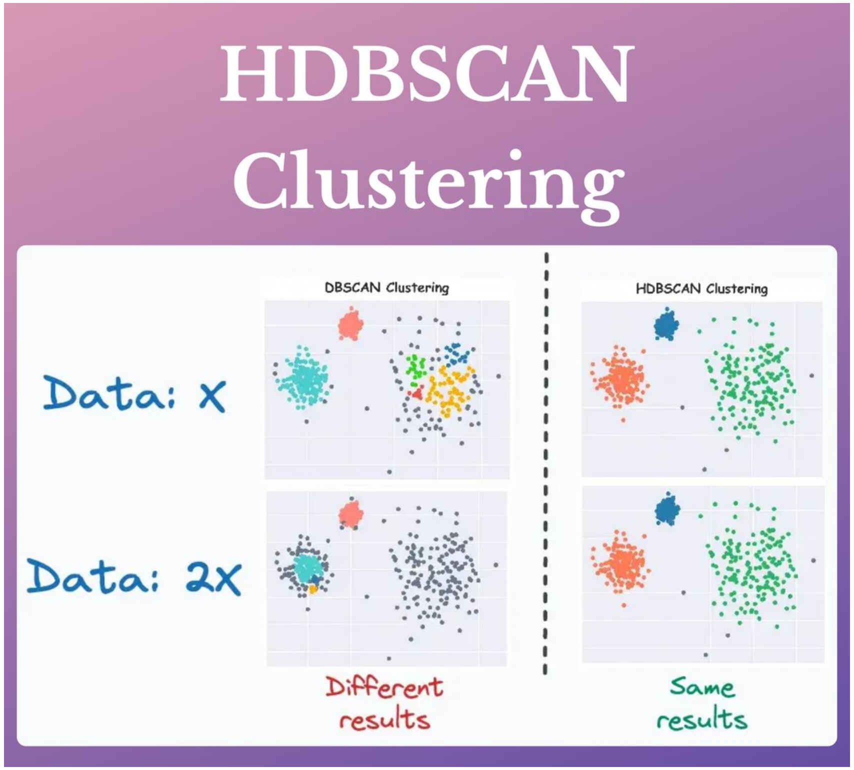
- DBSCAN – a common technique for clustering. Finds a cluster by looking for a group of points within a certain distance.



HDBSCAN

- HDBSCAN – does DBSCAN over different epsilon making the method more robust to scaling.





https://www.dailydoseofds.com/hdbscan-the-supercharged-version-of-dbscan-an-algorithmic-deep-dive/

Summary

- Text classification task
  - Bag of words model
    - Naïve Bayes
  - Neural based
- Text clustering
  - LDA
  - Top2vec