

Information Retrieval

CS 547/DS 547

Worcester Polytechnic Institute

Department of Computer Science

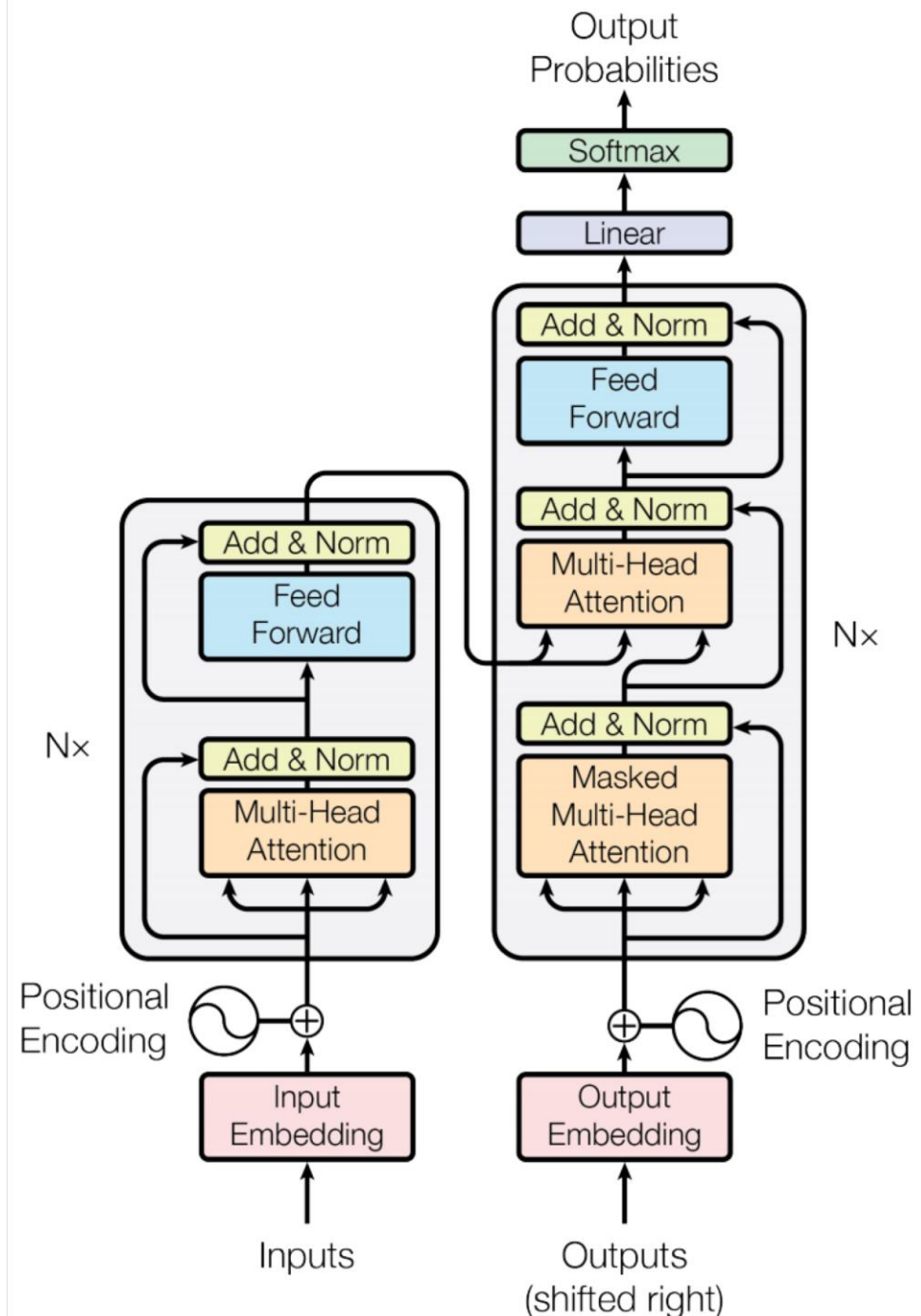
Instructor: Prof. Kyumin Lee

Transformer Overview

Attention is all you need. 2017. Aswani, Shazeer, Parmar, Uszkoreit, Jones, Gomez, Kaiser, Polosukhin <https://arxiv.org/pdf/1706.03762.pdf>

- Non-recurrent sequence-to-sequence encoder-decoder model
- Task: machine translation with parallel corpus
- Predict each translated word
- Final cost/error function is standard cross-entropy error on top of a softmax classifier

This and related figures from paper ↑



References for Transformer

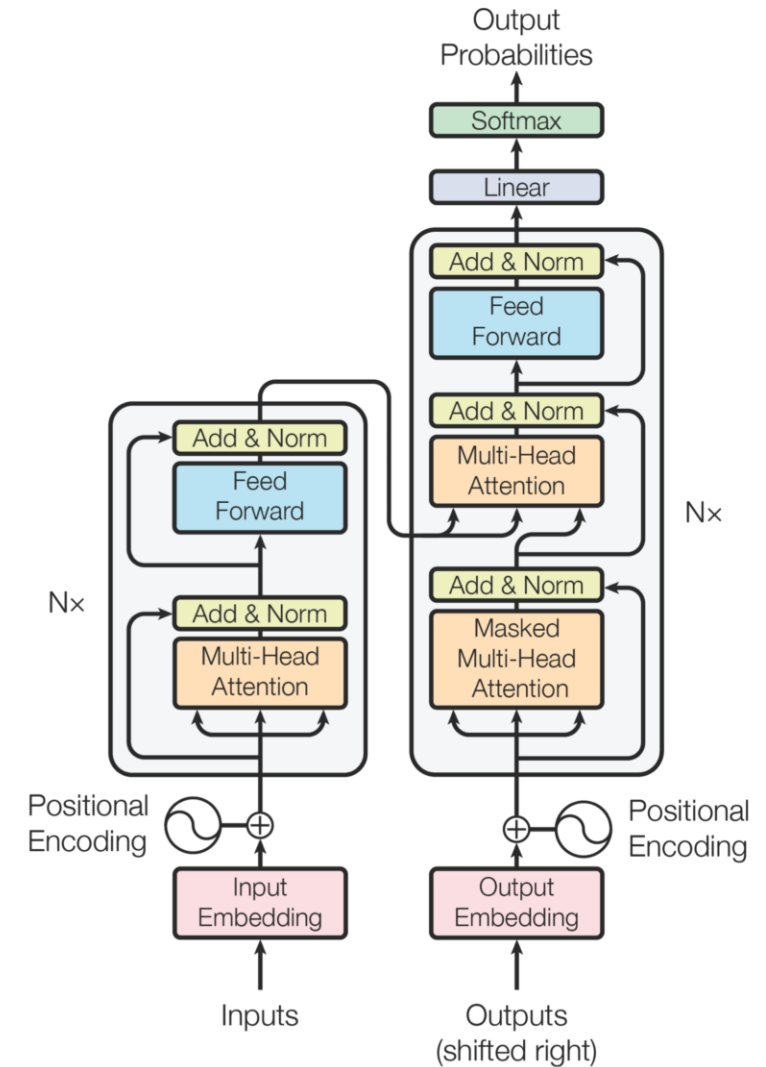


Figure 1: The Transformer - model architecture.

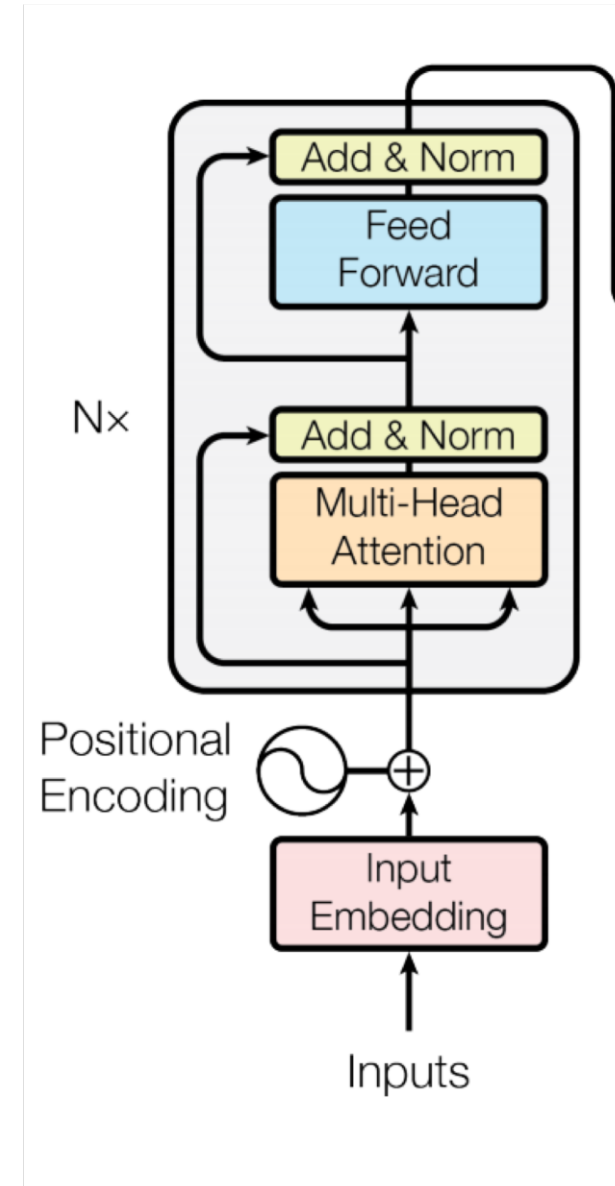
<http://jalammar.github.io/illustrated-transformer/>

<https://towardsdatascience.com/illustrated-guide-to-transformers-step-by-step-explanation-f74876522bc0>

Code: <https://www.tensorflow.org/text/tutorials/transformer>

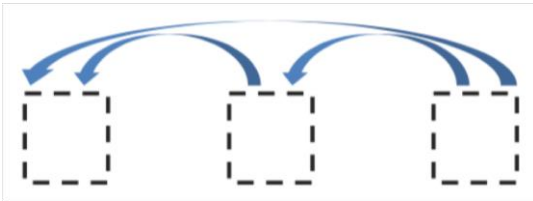
Transformer Encoder

- For encoder, at each block, we use the same Q, K and V from the previous layer
- Blocks are repeated 6 times (in vertical stack)

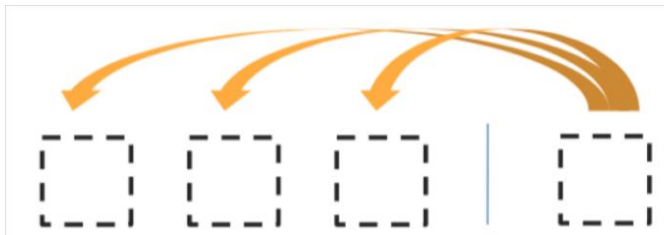


Transformer Decoder

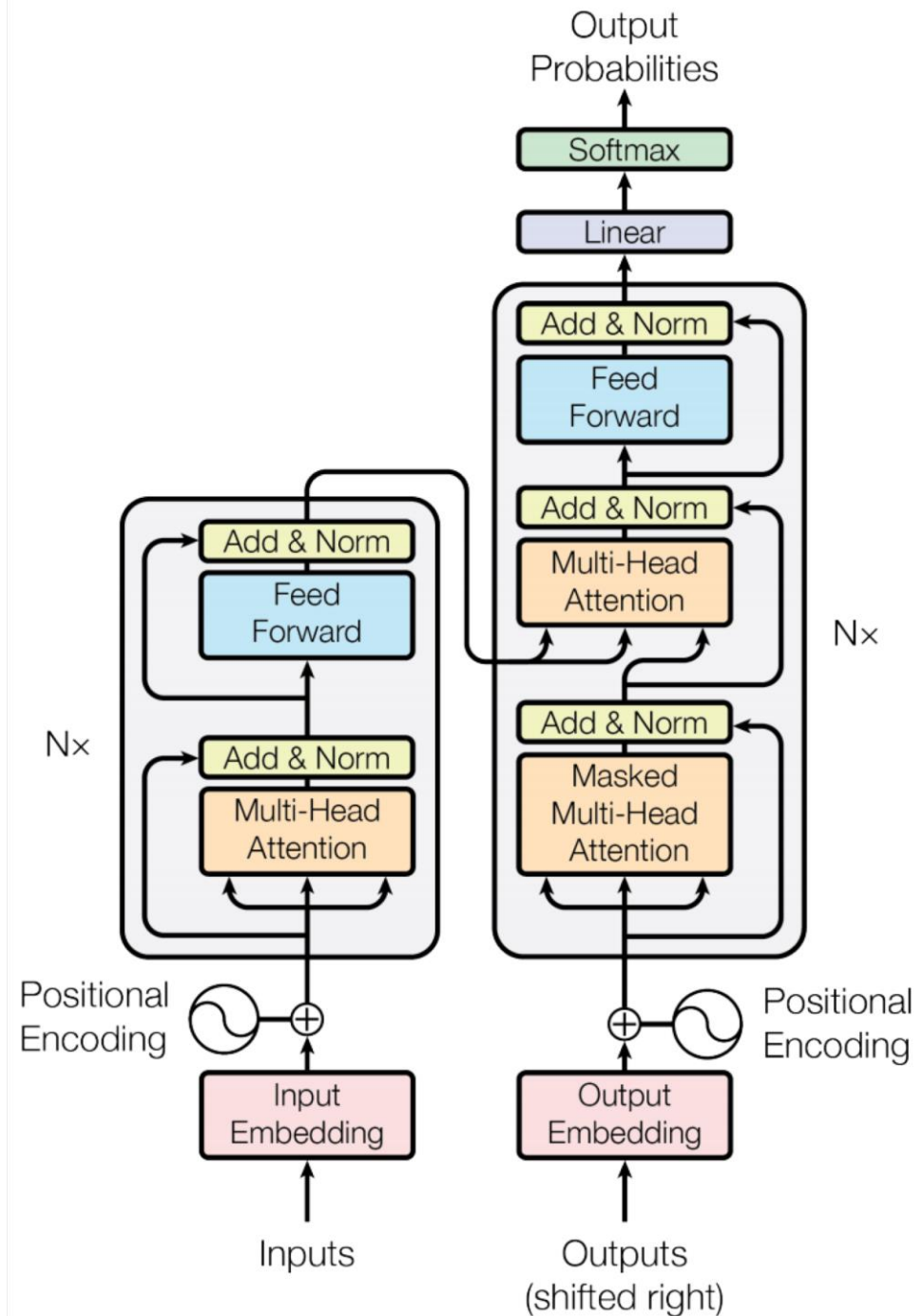
- 2 sublayer changes in decoder
- Masked decoder self-attention on previously generated outputs:



- Encoder-Decoder Attention, where queries come from previous decoder layer and keys and values come from output of encoder



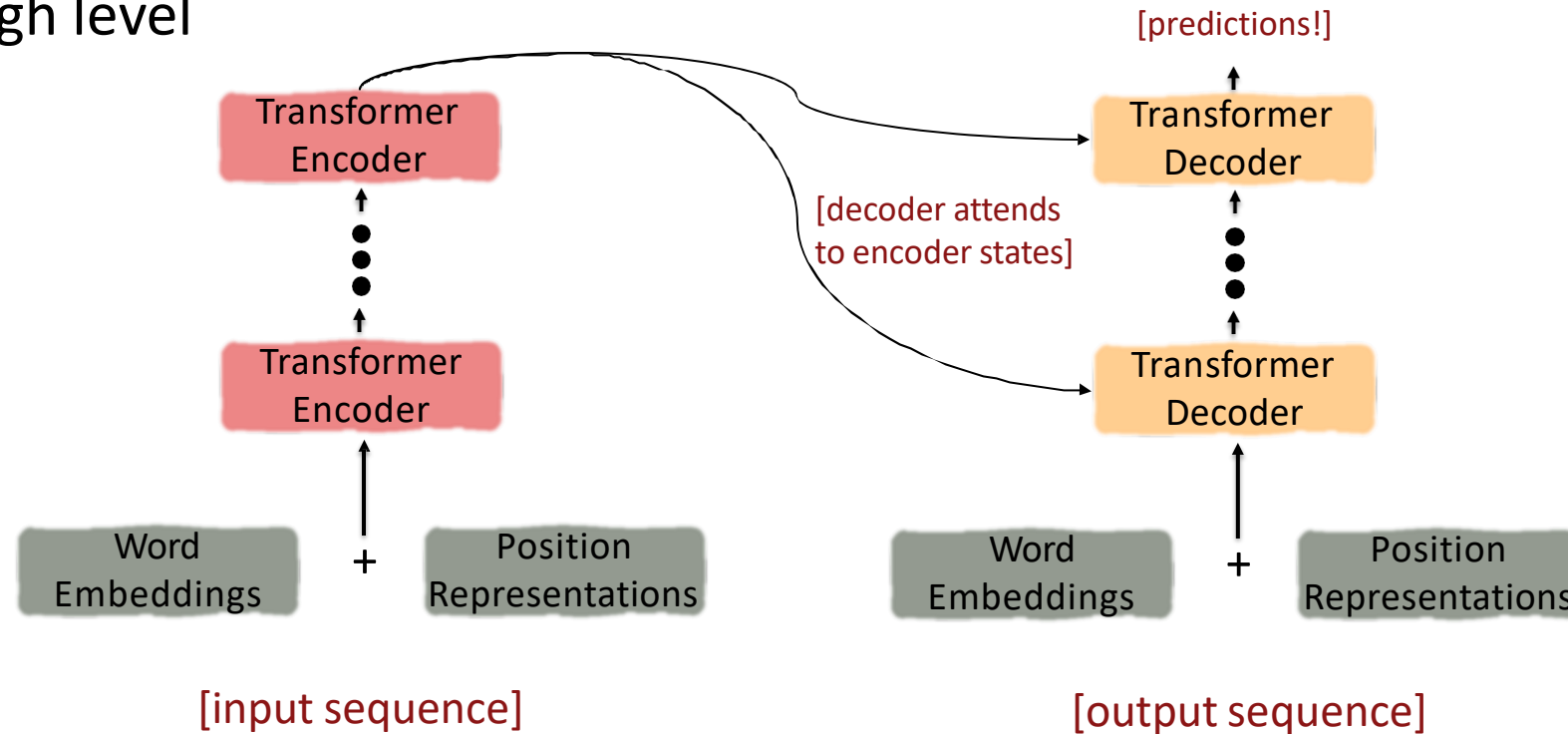
Blocks repeated 6 times also



The Transformer Encoder-Decoder

[Vaswani et al., 2017]

Another look at the Transformer Encoder and Decoder Blocks at a high level



The Transformer Encoder-Decoder

[Vaswani et al., 2017]

Next, let's look at the Transformer Encoder and Decoder Blocks

1. **Key-query-value attention:** How do we get the k, q, v vectors from a single word embedding?
2. **Multi-headed attention:** Attend to multiple places in a single layer!
3. **Tricks to help with training!**
 1. Scaling the dot product
 2. Residual connections
 3. Layer normalization
 4. These tricks **don't improve** what the model is able to do; they help improve the training process

The Transformer Encoder:

Dot-Product Attention

- Inputs: a query q and a set of key-value (k-v) pairs to an output
- Query, keys, values, and output are all vectors
- Output is weighted sum of values, where
- Weight of each value is computed by an inner product of query and corresponding key
- Queries and keys have same dimensionality d_k , value have d_v

$$A(q, K, V) = \sum_i \frac{e^{q \cdot k_i}}{\sum_j e^{q \cdot k_j}} v_i$$

The Transformer Encoder:

Dot-Product Attention – Matrix notation

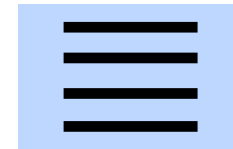
- When we have multiple queries q , we stack them in a matrix Q :

$$A(q, K, V) = \sum_i \frac{e^{q \cdot k_i}}{\sum_j e^{q \cdot k_j}} v_i$$

- Becomes: $A(Q, K, V) = \text{softmax}(QK^T)V$

$$[|Q| \times d_k] \times [d_k \times |K|] \times [|K| \times d_v]$$

softmax
row-wise



$$=[|Q| \times d_v]$$

The Transformer Encoder: Key-Query-Value Attention

- Self-attention is when keys, queries, and values come from the same source. The Transformer does this in a particular way:
 - Let x_1, \dots, x_T be input vectors to the Transformer encoder; $x_i \in \mathbb{R}^d$
- Then keys, queries, values are:
 - $k_i = Kx_i$, where $K \in \mathbb{R}^{d \times d}$ is the key matrix.
 - $q_i = Qx_i$, where $Q \in \mathbb{R}^{d \times d}$ is the query matrix.
 - $v_i = Vx_i$, where $V \in \mathbb{R}^{d \times d}$ is the value matrix.
- These matrices allow *different aspects* of the x vectors to be used/emphasized in each of the three roles.

The Transformer Encoder: Key-Query-Value Attention

- Let's look at how key-query-value attention is computed, in matrices.
 - Let $X = [x_1; \dots; x_T] \in \mathbb{R}^{T \times d}$ be the concatenation of input vectors.
 - First, note that $XK \in \mathbb{R}^{T \times d}$, $XQ \in \mathbb{R}^{T \times d}$, $XV \in \mathbb{R}^{T \times d}$.
 - The output is defined as $\text{output} = \text{softmax}(XQ(XK)^T) \times XV$.

First, take the query-key dot products in one matrix multiplication: $XQ(XK)$

$$XQ \cdot K^T X^T = XQK^T X^T \in \mathbb{R}^{T \times T}$$

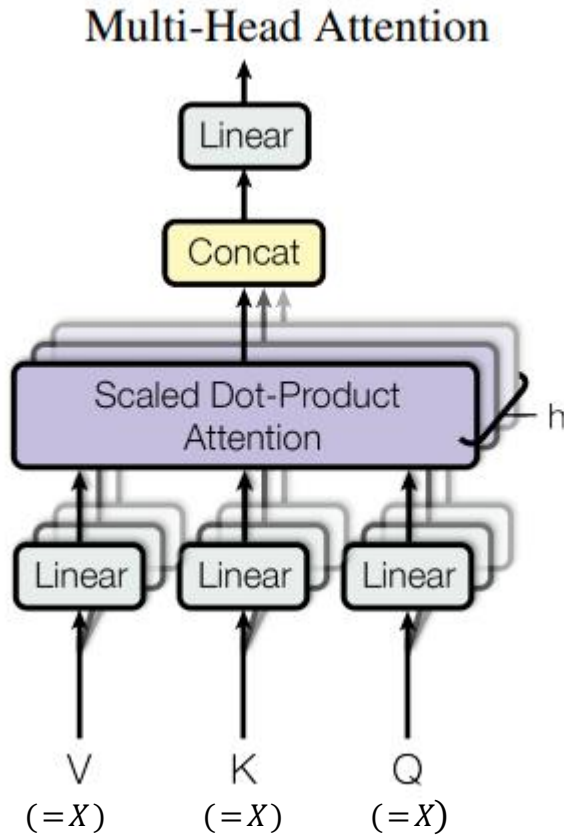
All pairs of attention scores!

Next, softmax, and compute the weighted average with another matrix multiplication.

$$\text{softmax}\left(XQK^T X^T\right) \cdot XV = \text{output} \in \mathbb{R}^{T \times d}$$

The Transformer Encoder:

Multi-headed attention



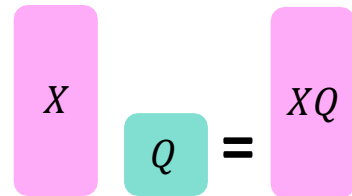
- What if we want to look in multiple places in the sentence at once?
 - For word i , self-attention “looks” where $x^T Q^T K x_j$ is high, but maybe we want to focus on different j for different reasons?
- We’ll define **multiple attention “heads”** through multiple Q,K,V matrices
- Let, $Q_P, K_P, V_P \in \mathbb{R}^{d \times \frac{d}{h}}$, where h is the number of attention heads, and P ranges from 1 to h .
- Each attention head performs attention independently:
 - $\text{output}_P = \text{softmax}(X Q_P K_P^T X^T) * X V_P$, where $\text{output}_P \in \mathbb{R}^{d/h}$
- Then the outputs of all the heads are combined!
 - $\text{output} = Y[\text{output}_1; \dots; \text{output}_h]$, where $Y \in \mathbb{R}^{d \times d}$
- Each head gets to “look” at different things, and construct value vectors differently.

The Transformer Encoder:

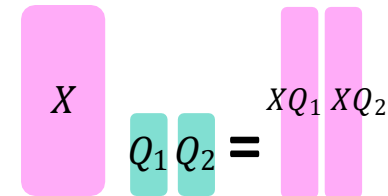
Multi-headed attention

- What if we want to look in multiple places in the sentence at once?
 - For word i , self-attention “looks” where $x^T Q^T K x_j$ is high, but maybe we want to focus on different j for different reasons?
- We’ll define **multiple attention “heads”** through multiple Q,K,V matrices
- Let, $Q_P, K_P, V_P \in \mathbb{R}^{d \times \frac{d}{h}}$, where h is the number of attention heads, and P ranges from 1 to h .

Single-head attention
(just the query matrix)



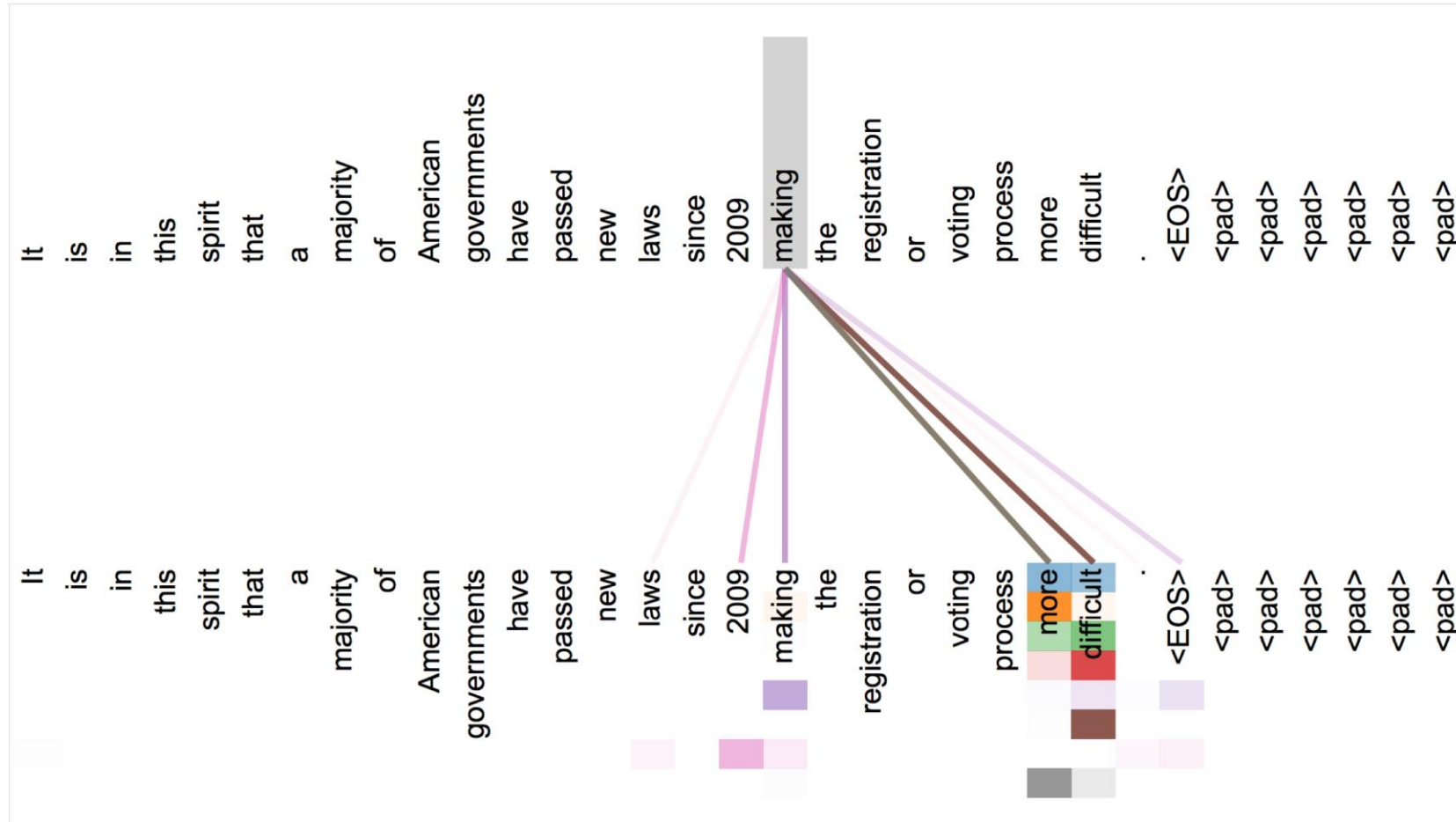
Multi-head attention
(just two heads here)



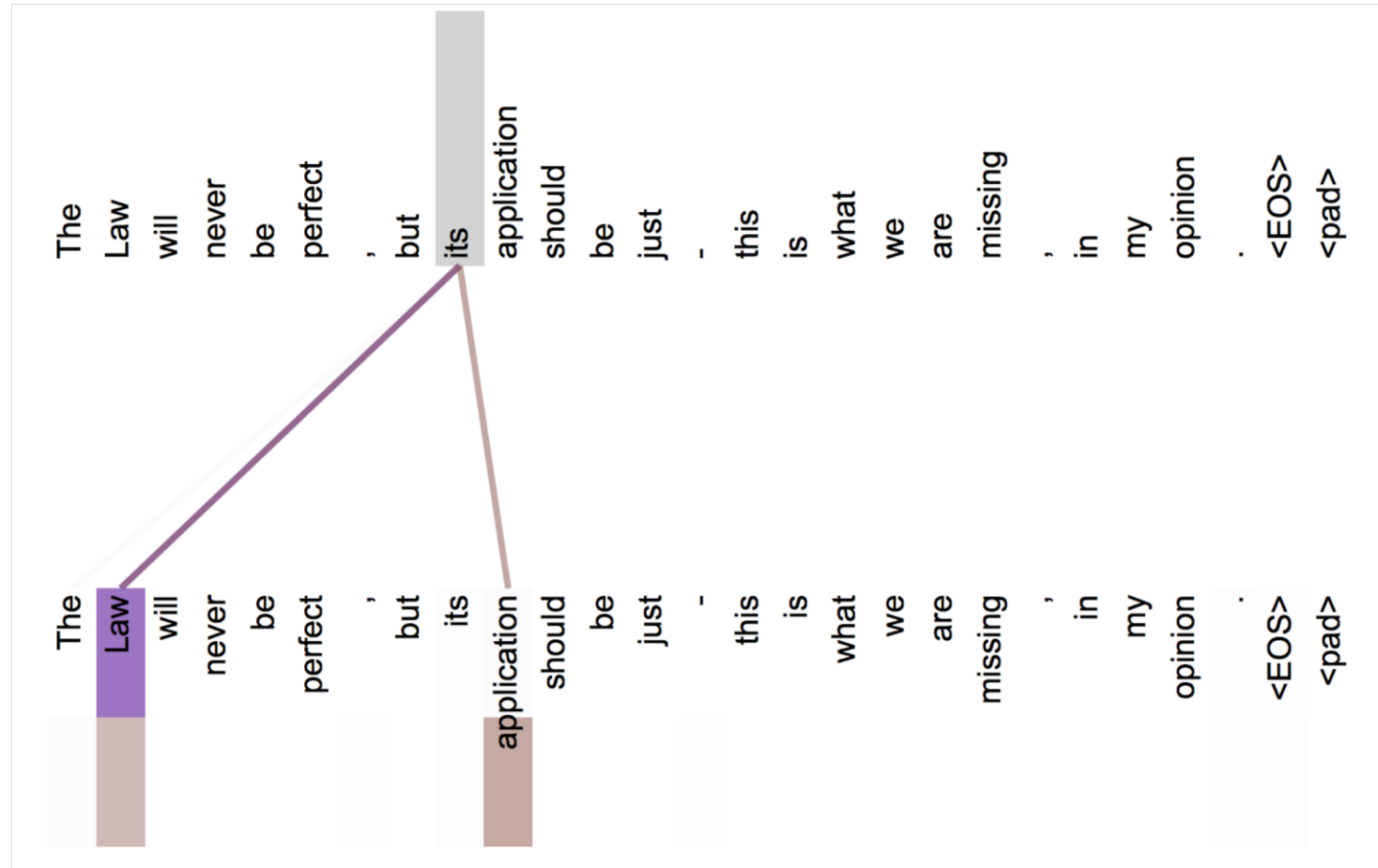
Same amount of
computation as
single-head self-
attention!

Attention visualization in layer 5

- Words start to pay attention to other words in sensible ways



Attention visualization: Implicit anaphora resolution

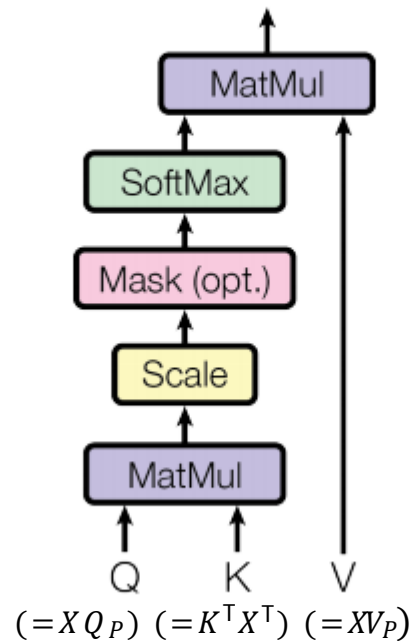


In 5th layer. Isolated attentions from just the word 'its' for attention heads 5 and 6.
Note that the attentions are very sharp for this word.

The Transformer Encoder:

Scaled Dot Product [\[Vaswani et al., 2017\]](#)

Scaled Dot-Product Attention



- “**Scaled Dot Product**” attention is a final variation to aid in Transformer training.
- When dimensionality d becomes large, dot products between vectors become large, inputs to the softmax function can be large, making gradients small.

- Instead of the self-attention function we’ve seen:

$$\text{output}_P = \text{softmax}(XQ_P K_P^T X^T) * XV_P$$

- We divide the attention scores by $\sqrt{d/h}$, to stop the scores from becoming large just as a function of d/h (The dimensionality divided by the number of heads.)

$$\text{output}_P = \text{softmax}\left(\frac{XQ_P K_P^T X^T}{\sqrt{d/h}}\right) * XV_P$$

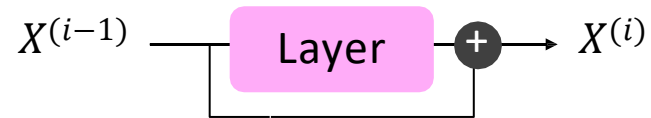
The Transformer Encoder:

Residual connections [\[He et al., 2016\]](#)

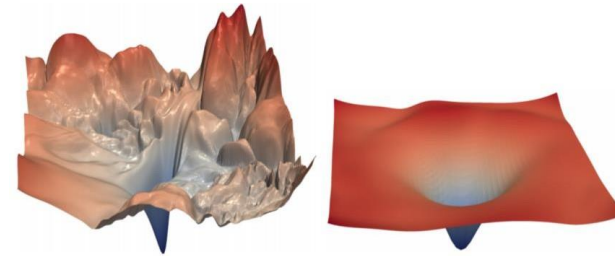
- **Residual connections** are a trick to help models train better.
 - Instead of $X^{(i)} = \text{Layer}(X^{(i-1)})$ (where i represents the layer)



- We let $X^{(i)} = X^{(i-1)} + \text{Layer}(X^{(i-1)})$ (so we only have to learn “the residual” from the previous layer)



- Residual connections are thought to make the loss landscape considerably smoother (thus easier training!)



[no residuals]

[residuals]

[Loss landscape visualization,
[Li et al., 2018](#), on a ResNet]

The Transformer Encoder:

Layer normalization [[Ba et al., 2016](#)]

- **Layer normalization** is a trick to help models train faster.
- Idea: cut down on uninformative variation in hidden vector values by normalizing to unit mean and standard deviation **within each layer**.
 - LayerNorm's success may be due to its normalizing gradients [[Xu et al., 2019](#)]
- Let $x \in \mathbb{R}^d$ be an individual (word) vector in the model.
- Let $\mu = \frac{1}{d} \sum_{j=1}^d x_j$; this is the mean; $\mu \in \mathbb{R}$.
 - Let $\sigma = \sqrt{\frac{1}{d} \sum_{j=1}^d (x_j - \mu)^2}$; this is the standard deviation; $\sigma \in \mathbb{R}$.
 - Let $\gamma \in \mathbb{R}^d$ and $\beta \in \mathbb{R}^d$ be learned “gain” and “bias” parameters. (Can omit!)
 - Then layer normalization computes:

$$\text{output} = \frac{x - \mu}{\sqrt{\sigma^2 + \epsilon}} * \gamma + \beta$$

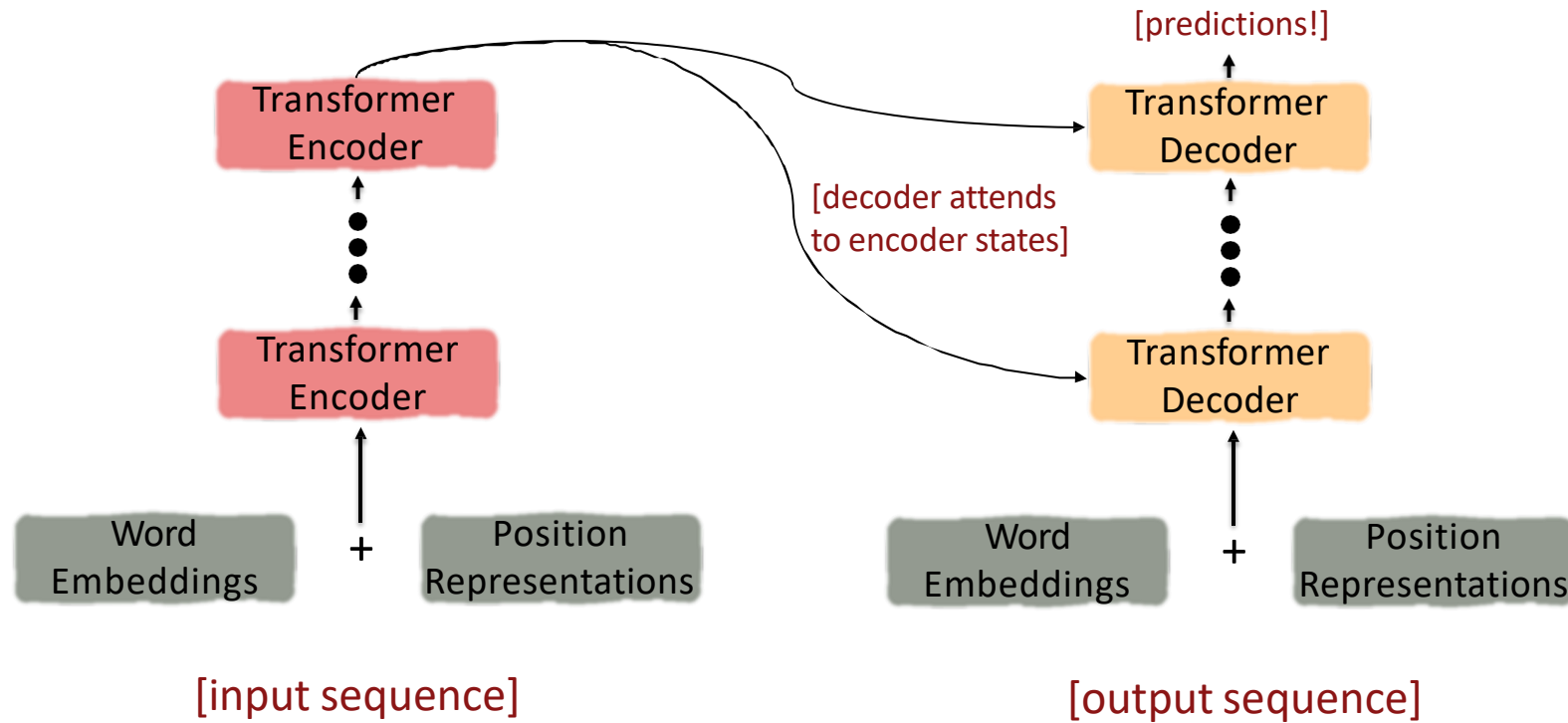
Normalize by scalar mean and variance

Modulate by learned elementwise gain and bias

The Transformer Encoder-Decoder

[Vaswani et al., 2017]

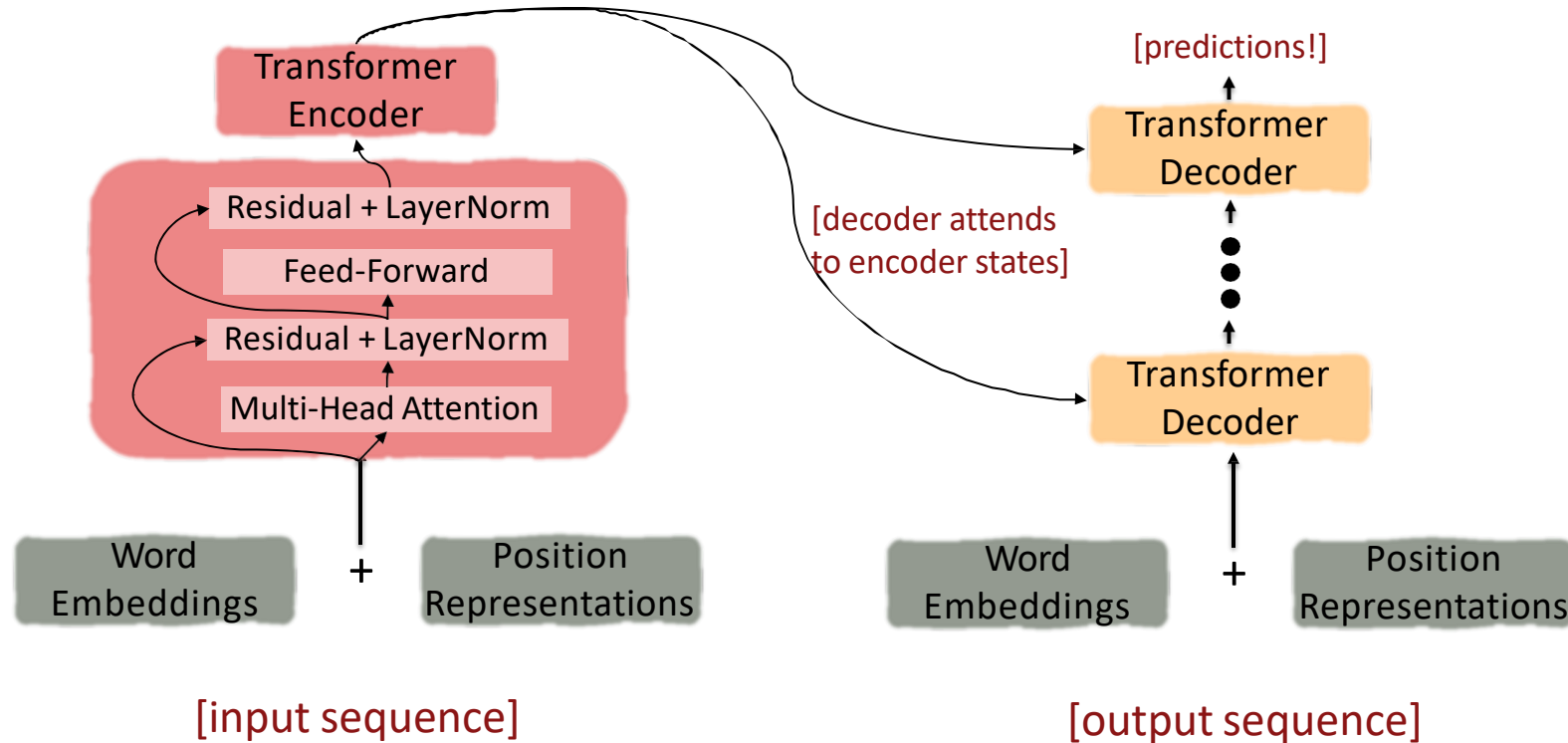
Looking back at the whole model, zooming in on an Encoder block:



The Transformer Encoder-Decoder

[Vaswani et al., 2017]

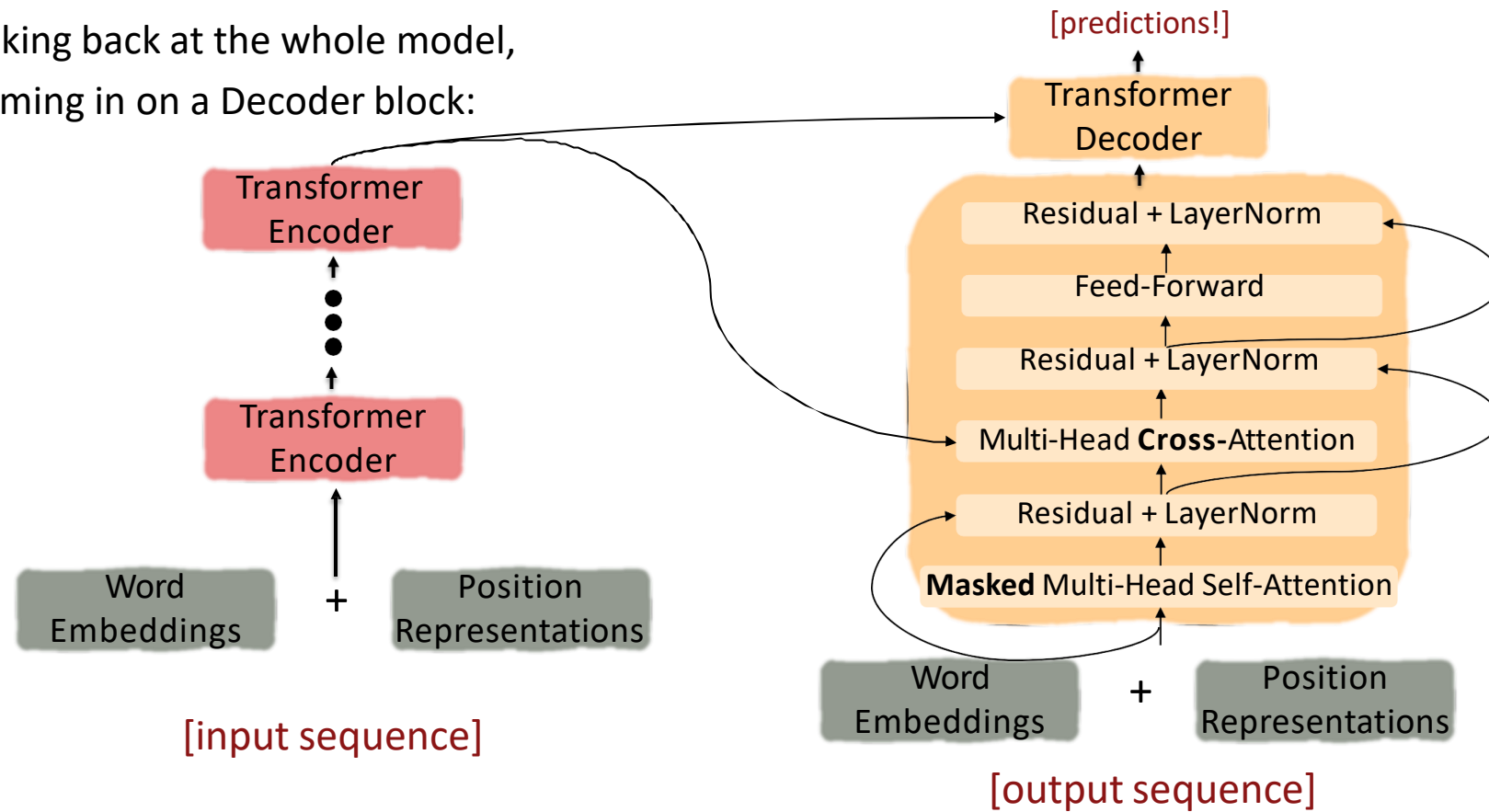
Looking back at the whole model, zooming in on an Encoder block:



The Transformer Encoder-Decoder

[Vaswani et al., 2017]

Looking back at the whole model,
zooming in on a Decoder block:

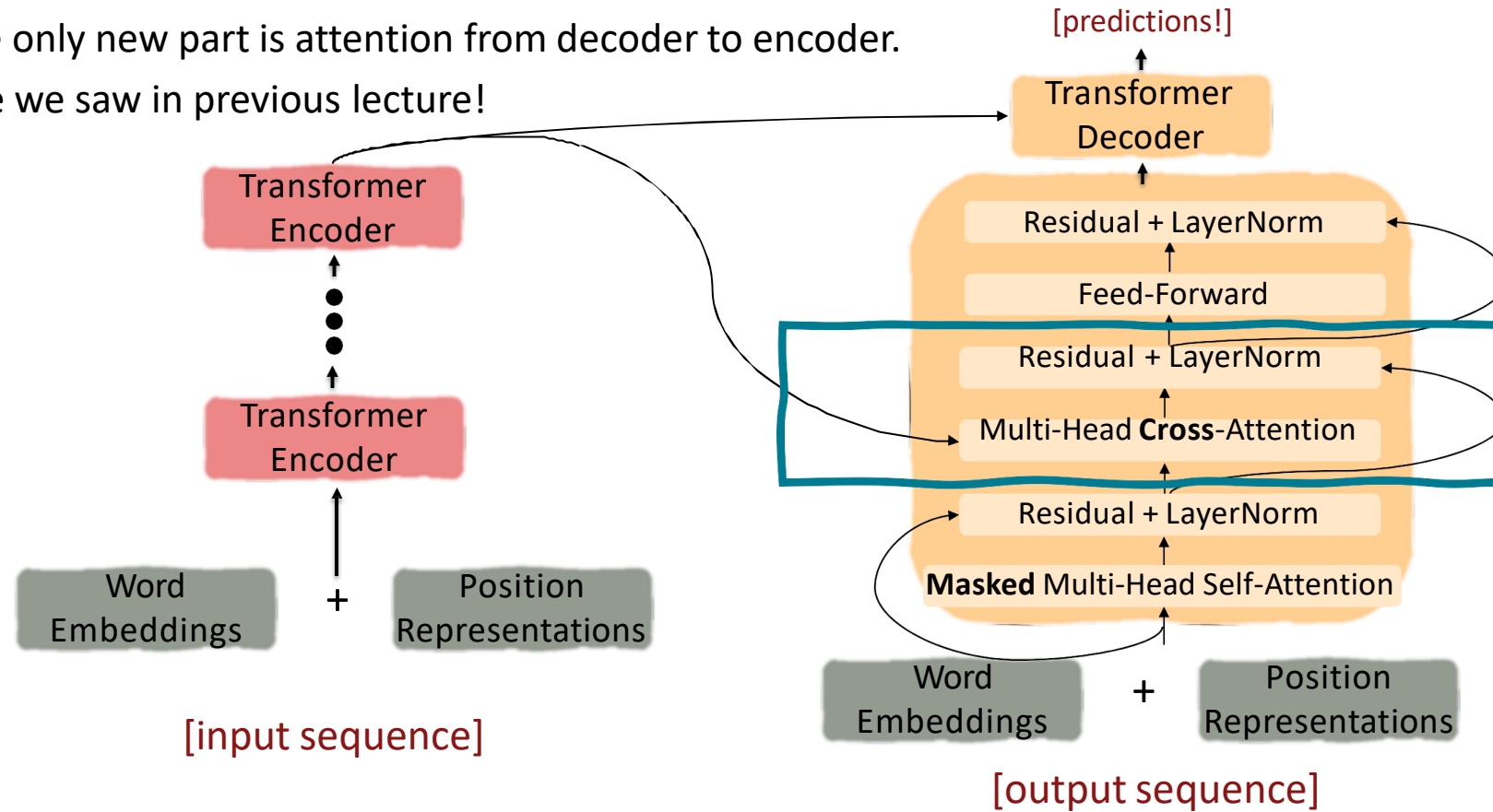


The Transformer Encoder-Decoder

[Vaswani et al., 2017]

The only new part is attention from decoder to encoder.

Like we saw in previous lecture!



The Transformer Decoder: Cross-attention (details)

- We saw self-attention is when keys, queries, and values come from the same source.
- Let h_1, \dots, h_T be **output** vectors from the Transformer **encoder**; $x_i \in \mathbb{R}^d$
- Let z_1, \dots, z_T be input vectors from the Transformer **decoder**, $z_i \in \mathbb{R}^d$
- Then keys and values are drawn from the **encoder** (like a memory):
 - $k_i = Kh_i, v_i = Vh_i$.
- And the queries are drawn from the **decoder**, $q_i = Qz_i$.

The Transformer Encoder: Cross-attention (details)

- Let's look at how cross-attention is computed, in matrices.
 - Let $H = [h_1; \dots; h_T] \in \mathbb{R}^{T \times d}$ be the concatenation of encoder vectors.
 - Let $Z = [z_1; \dots; z_T] \in \mathbb{R}^{T \times d}$ be the concatenation of decoder vectors.
 - The output is defined as $\text{output} = \text{softmax}(ZQ(HK)) \times HV$.

First, take the query-key dot products in one matrix multiplication: $ZQ(HK)$

A diagram illustrating the first step of cross-attention. It shows a pink vertical rectangle labeled ZQ multiplied by an orange horizontal rectangle labeled $K^T H^T$. The result is a brown rounded rectangle labeled $ZQK^T H^T$. To the right of this rectangle is the text $\in \mathbb{R}^{T \times T}$. A blue text label "All pairs of attention scores!" points to the brown rectangle.

$$ZQ \cdot K^T H^T = ZQK^T H^T \in \mathbb{R}^{T \times T}$$

Next, softmax, and compute the weighted average with another matrix multiplication.

A diagram illustrating the second step of cross-attention. It shows a brown rounded rectangle labeled $ZQK^T H^T$ enclosed in large square brackets. To the left of the brackets is the word "softmax". To the right of the brackets is an orange vertical rectangle labeled HV . This is followed by an equals sign and another orange vertical rectangle. To the right of this final rectangle is the text "output $\in \mathbb{R}^{T \times d}$ ". An arrow points from the brown rectangle in the first equation to the brown rectangle in this equation.

$$\text{softmax} \left(ZQK^T H^T \right) \cdot HV = \text{output} \in \mathbb{R}^{T \times d}$$

References for Transformer

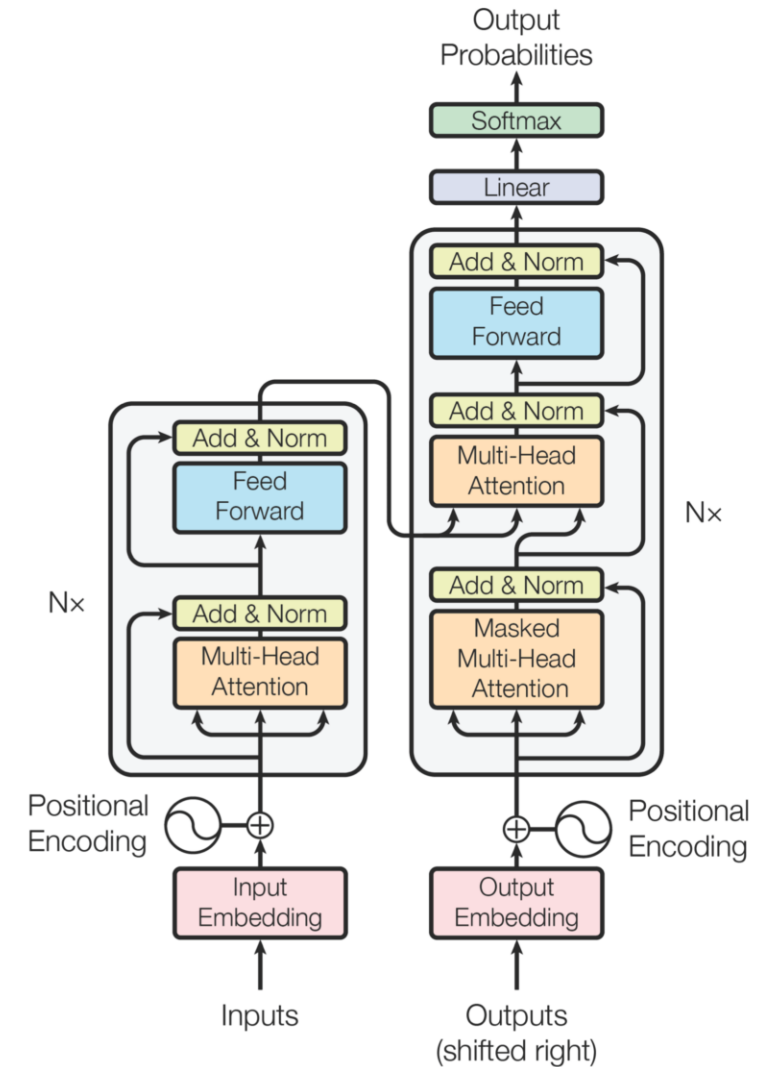


Figure 1: The Transformer - model architecture.

<http://jalammar.github.io/illustrated-transformer/>

<https://towardsdatascience.com/illustrated-guide-to-transformers-step-by-step-explanation-f74876522bc0>

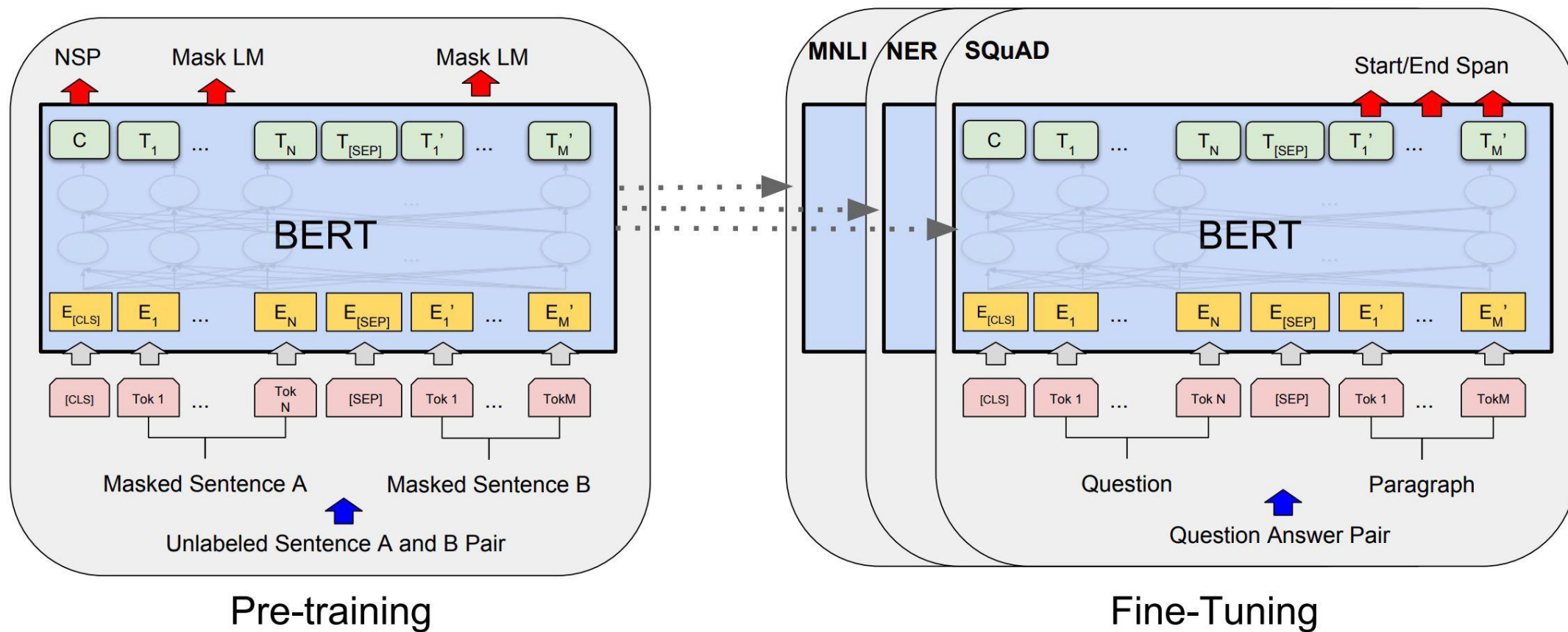
Code: <https://www.tensorflow.org/text/tutorials/transformer>

An example of Transformer-based Classification

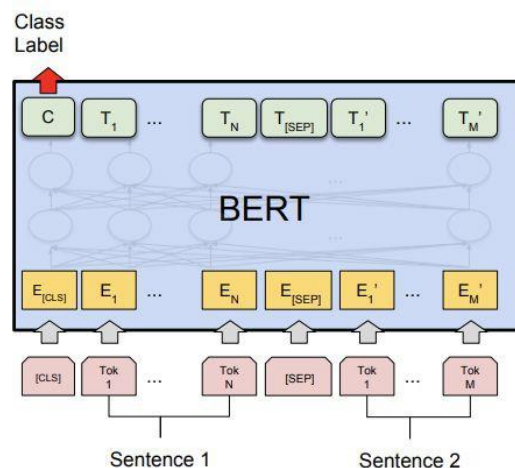
- On top of Transformer Encoder Block
- Add GlobalAveragePooling
- Add Feedforward Neural Net
- Then, do softmax
- https://keras.io/examples/nlp/text_classification_with_transformer/

BERT

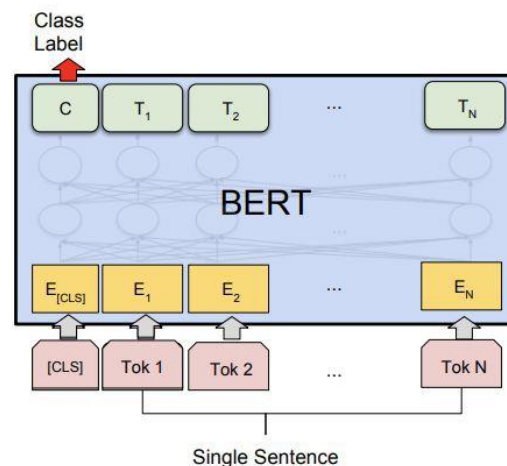
Fine-Tuning Procedure



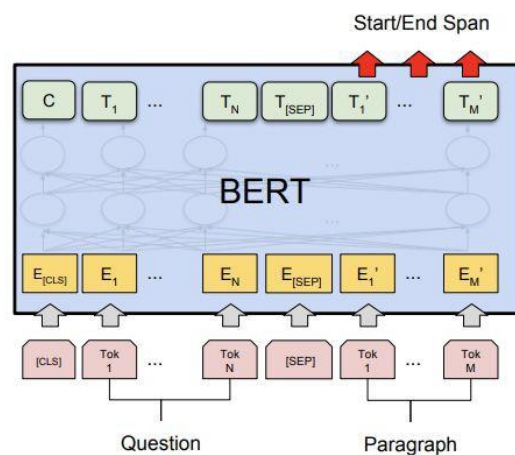
Fine-Tuning Procedure



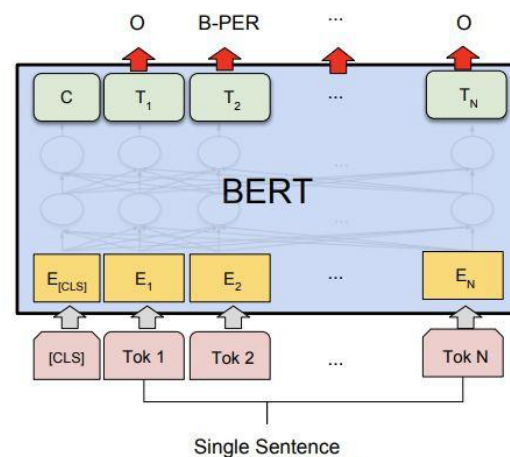
(a) Sentence Pair Classification Tasks:
MNLI, QQP, QNLI, STS-B, MRPC,
RTE, SWAG



(b) Single Sentence Classification Tasks:
SST-2, CoLA



(c) Question Answering Tasks:
SQuAD v1.1



(d) Single Sentence Tagging Tasks:
CoNLL-2003 NER

BERT was released in 2018

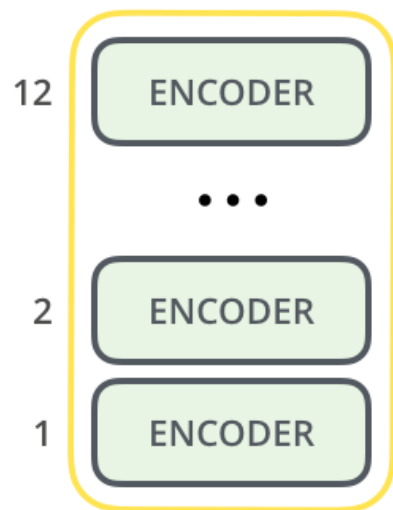
BERT_{BASE} (L=12, H=768, A=12, Total Parameters=110M)

BERT_{LARGE} (L=24, H=1024, A=16, Total Parameters=340M)

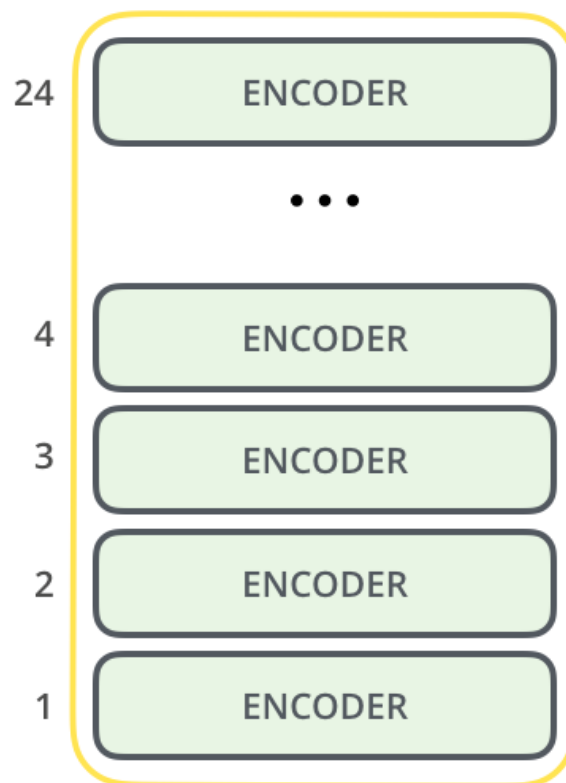
L = the number of layers;

H = hidden size

A = number of attention head



BERT_{BASE}



BERT_{LARGE}

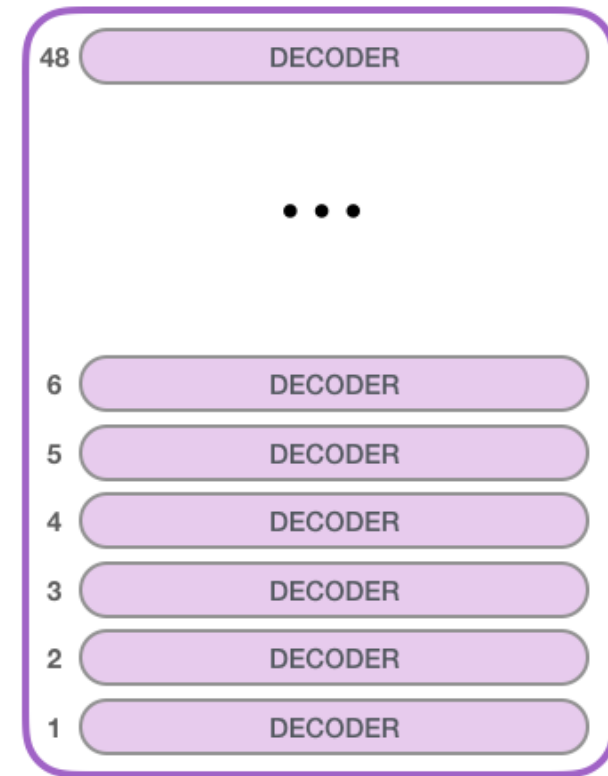
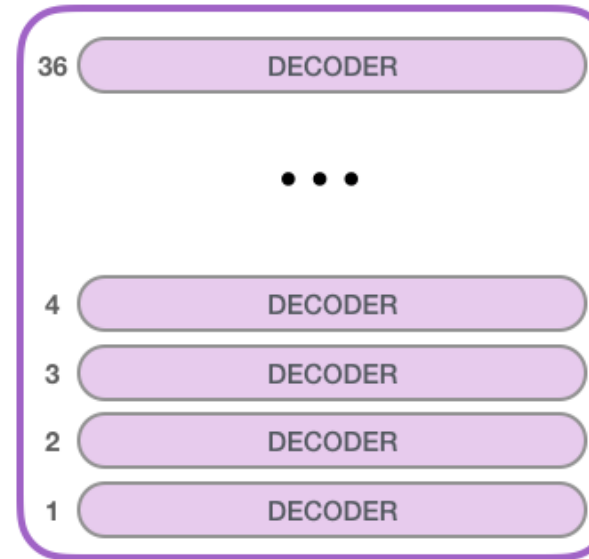
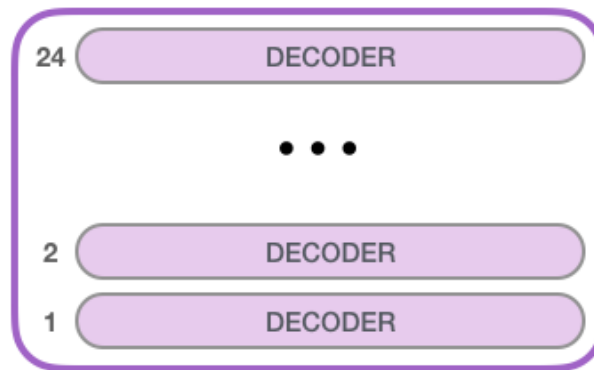
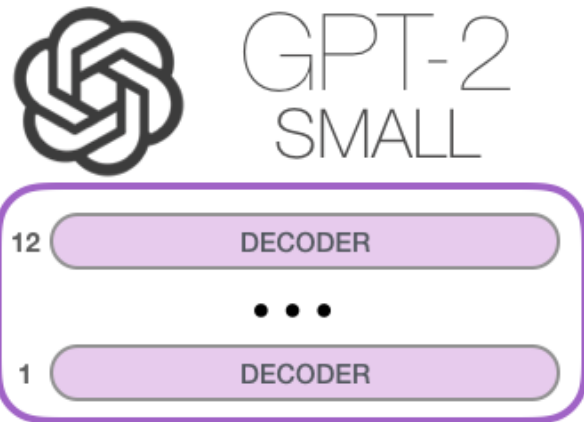
GPT was released in 2018

GPT-2 was released in 2019 with 1.5B parameters

GPT-3 was released in 2020 with 175B parameters

GPT-3.5 was released in 2022 with 355B parameters

GPT-4 was released in 2023 with 1T parameters



Model Dimensionality: 768

Model Dimensionality: 1024

Model Dimensionality: 1280

Model Dimensionality: 1600

117M parameters

345M

762M

1542M

Evaluating a Classifier

Evaluating classification

- Evaluation must be done on test data that are independent of the training data (usually a disjoint set of instances).
- It's easy to get good performance on a test set that was available to the learner during training (e.g., just memorize the test set).
- Measures: Precision, recall, F_1 , classification accuracy

Precision P and recall R

	in the class	not in the class
predicted to be in the class	true positives (TP)	false positives (FP)
predicted to not be in the class	false negatives (FN)	true negatives (TN)

$$P = TP / (TP + FP)$$

$$R = TP / (TP + FN)$$

A combined measure: F

- F_1 allows us to trade off precision against recall.

- $$F_1 = \frac{1}{\frac{1}{2}\frac{1}{P} + \frac{1}{2}\frac{1}{R}} = \frac{2PR}{P + R}$$

- This is the **harmonic mean** of P and R : $\frac{1}{F} = \frac{1}{2}\left(\frac{1}{P} + \frac{1}{R}\right)$

Averaging: Micro vs. Macro

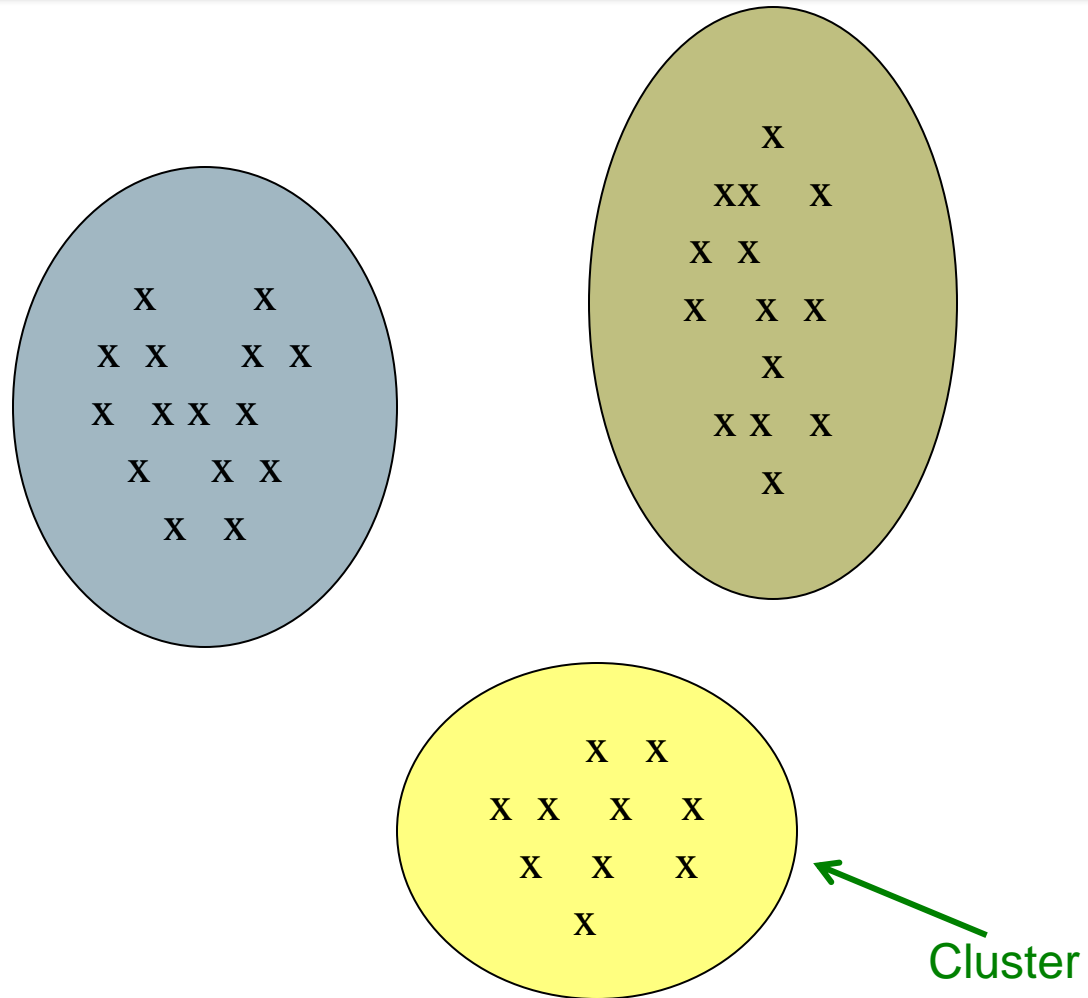
- We now have an evaluation measure (F_1) for **one class**.
- But we also want a single number that measures the **aggregate performance** over all classes in the collection.
- **Macroaveraging**
 - Compute F_1 for each of the C classes
 - Average these C numbers
- **Microaveraging**
 - Compute TP, FP, FN for each of the C classes
 - Sum these C numbers (e.g., all TP to get aggregate TP)
 - Compute F_1 for aggregate TP, FP, FN

Clustering

What is Clustering?

- **Clustering** is the process of grouping a set of documents into clusters of similar documents.
 - Documents within a cluster should be similar.
 - Documents from different clusters should be dissimilar.
- Clustering is the most common form of *unsupervised learning*.
 - Unsupervised learning = learning from raw data, as opposed to supervised data where a classification of examples is given
- A common and important task that finds many applications in IR and other places

Example Clusters



Clustering is Hard!



Why is it hard?

- Clustering in two dimensions looks easy
- Clustering small amounts of data looks easy
- And in most cases, looks are **not** deceiving

- Many applications involve not 2, but 10 or 10,000 dimensions
- **High-dimensional spaces look different:** Almost all pairs of points are at about the same distance



- Typical applications
 - As a **stand-alone tool** to get insight into data distribution
 - As a **preprocessing step** for other algorithms

Clustering in IR

Applications of clustering in IR

- **Whole corpus analysis/navigation**
 - **Better user interface: search without typing**
- For improving recall in search applications
 - Better search results
- For better navigation of search results
 - Effective “user recall” will be higher

Google News: automatic clustering gives an effective news presentation metaphor




News

U.S. editionModern

Top Stories

- Bernie Sanders
- Denver Broncos
- Syria
- Trans-Pacific Partnership
- Jeb Bush
- Dave Mirra
- Roger Goodell
- Manhattan
- OPEC
- Flint
- Utah
- World
- U.S.
- Business
- Technology
- Entertainment
- Sports
- Science
- Health

Top Stories



Los Angeles Times

Taiwan earthquake topples buildings, leaving at least 7 dead and hundreds injured

Los Angeles Times - 46 minutes ago

Several buildings collapsed when a magnitude 6.4 earthquake struck before dawn in southern Taiwan on Feb. 6, 2016. Julie Makinen, Samuel Chan and Jonathan Kaiman Contact Reporters.




[7 Dead, Hundreds Rescued and Injured as Quake Rattles Taiwan](#) ABC News

[6.4-Magnitude Earthquake Strikes Taiwan](#) NBCNews.com


From Taiwan: [1999 quake survivor rescued from toppled building in Taiwan](#) temblor

[Wikipedia: 2016 Kaohsiung earthquake](#)

[See realtime coverage](#)




YouTubeYouTubeThe Star Online



Washington Post - 8 hours ago

Clinton, Sanders use NH primary to frame long battle to come


CONCORD, N.H. - For the Democratic presidential candidates, there are two urgent campaigns underway in New Hampshire. The first is over the size of what Hillary Clinton and Bernie Sanders agree is a likely Sanders victory here: Clinton is pulling out ...



New York Times - 5 hours ago

Crane Collapse in Lower Manhattan Kills One Person


The crew operating a crane in Lower Manhattan on Friday morning took note of the wind gusts accompanying the falling snow. The workers, officials said, decided they needed to lower the crane to a secure level, and so around 8 a.m.



Tulsa World - 41 minutes ago

Twitter moves to actively seek out terrorist supporters

WASHINGTON - Twitter is now using spam-fighting technology to seek out accounts that might be promoting terrorist activity and is examining other accounts related to those flagged for possible removal, the company announced Friday.



seattlepi.com - 41 minutes ago

3 people believed aboard 2 planes that collided off LA

Related

[Southern Taiwan »](#)

[Taiwan »](#)

Applications of clustering in IR

- Whole corpus analysis/navigation
 - Better user interface: search without typing
- **For improving recall in search applications**
 - **Better search results**
- For better navigation of search results
 - Effective “user recall” will be higher

For improving search recall

- *Cluster hypothesis* - Documents in the same cluster behave similarly with respect to relevance to information needs
- Therefore, to improve search recall:
 - Cluster docs in corpus a priori
 - When a query matches a doc D , also return other docs in the cluster containing D
- Hope if we do this: The query “car” will also return docs containing *automobile*
 - Because clustering grouped together docs containing *car* with those containing *automobile*.

Applications of clustering in IR

- Whole corpus analysis/navigation
 - Better user interface: search without typing
- For improving recall in search applications
 - Better search results
- **For better navigation of search results**
 - **Effective “user recall” will be higher**

Yippy.com (no longer active)

new.yippy.com/search?input-form=clusty-simple&v%3Asources=webplus-ns-aaf&v%3Aproject=clusty-new&query=aggies

web news wikipedia jobs more »





aggies Search advanced preferences





clouds sources sites time remix





All Results (232)





- + Roster, Schedule (69)
- + College Station (24)
- + Athletics (21)
- + Texas Tech Red Raiders (20)
- + Baseball (12)
- + College Football (9)
- + Dallas (8)
- + SEC (7)
- + Tickets (7)
- + Star (7)
- Southeastern Conference (3)
- Reviews (3)
- Blogs, News (3)
- Connection (4)
- Www.statesman.com (3)
- Serves (3)
- Dictionary (2)
- Land grant university (2)
- TexAgs, M Football (2)
- Hurricanes (2)
- Aggies continually updated from thousands of sources (2)
- Apparel & Merchandise (2)





Top 232 results of at least 4,390,000 retrieved for the query **aggies** (details)





[New Mexico State University Athletics](#)    
Official site of the **Aggies** with news, schedules and live audio.
[www.nmstatesports.com](#) - [cache] - Yippy Index I, Yippy Index IV





[Texas A&M University Athletics](#)    
Official site of the Texas A&M Athletic Department. Schedules and ticket information for all sporting events.
[www.12thman.com](#) - [cache] - Yippy Index IV

[Utah State Aggies](#)    
The Official Athletic Site of the Utah State **Aggies**, partner of CBSSports.com College Network. The most comprehensive coverage of Utah State Athletics on the web.
[www.utahstateaggies.com](#) - [cache] - Yippy Index IV, Yippy Index I

[Home - Texas A&M University, College Station, TX](#)    
The oldest public university in Texas, this flagship university provides the best return-on-investment among Texas's public schools, with more than 400 degrees.
[www.tamu.edu](#) - [cache] - Yippy Index IV

[Stadium Journey - Stadium Reviews and Sports Travel Community](#)    
... Bay Rowdies Tampa Bay Storm Texas A&M **Aggies** Softball Texas Rangers Spring Training The Highlanders The ... Hampshire Wildcats New Mexico Lobos New Mexico State **Aggies** Norfolk State Spartans North Carolina A&T **Aggies** ...
[www.stadiumjourney.com](#) - [cache] - Yippy Index

[Texas A&M Aggies - Wikipedia, the free encyclopedia](#)    
Texas A&M **Aggies** (variously A&M or Texas **Aggies**) refers to the students, graduates, and sports teams of Texas A&M University. The nickname "**Aggie**" was once common at ...
[https://en.wikipedia.org/wiki/Texas_A&M_Aggies](#) - [cache] - Yippy Index IV

[Aggies land four-star running back prospect Trayveon Williams | Fox News](#)  
 
Nov 12, 2015 Aggies land four star running back prospect Trayveon Williams

Issues for clustering

- Representation for clustering
 - Document representation
 - Vector space? Normalization?
 - Need a notion of similarity/distance
- How many clusters?
 - Fixed a priori?
 - Completely data driven?
 - Avoid “trivial” clusters - too large or small
 - If a cluster's too large, then for navigation purposes you've wasted an extra user click without whittling down the set of documents much.

Flat vs. Hierarchical Clustering

- Flat algorithms
 - Usually start with a random (partial) partitioning
 - Refine it iteratively
 - Main algorithm: K-means
- Hierarchical algorithms
 - Create a hierarchy
 - Bottom-up, agglomerative
 - Start with all documents belong to the same cluster.
 - Eventually each node forms a cluster on its own.
 - Top-down, divisive
 - Start with each document being a single cluster.
 - Eventually all documents belong to the same cluster.

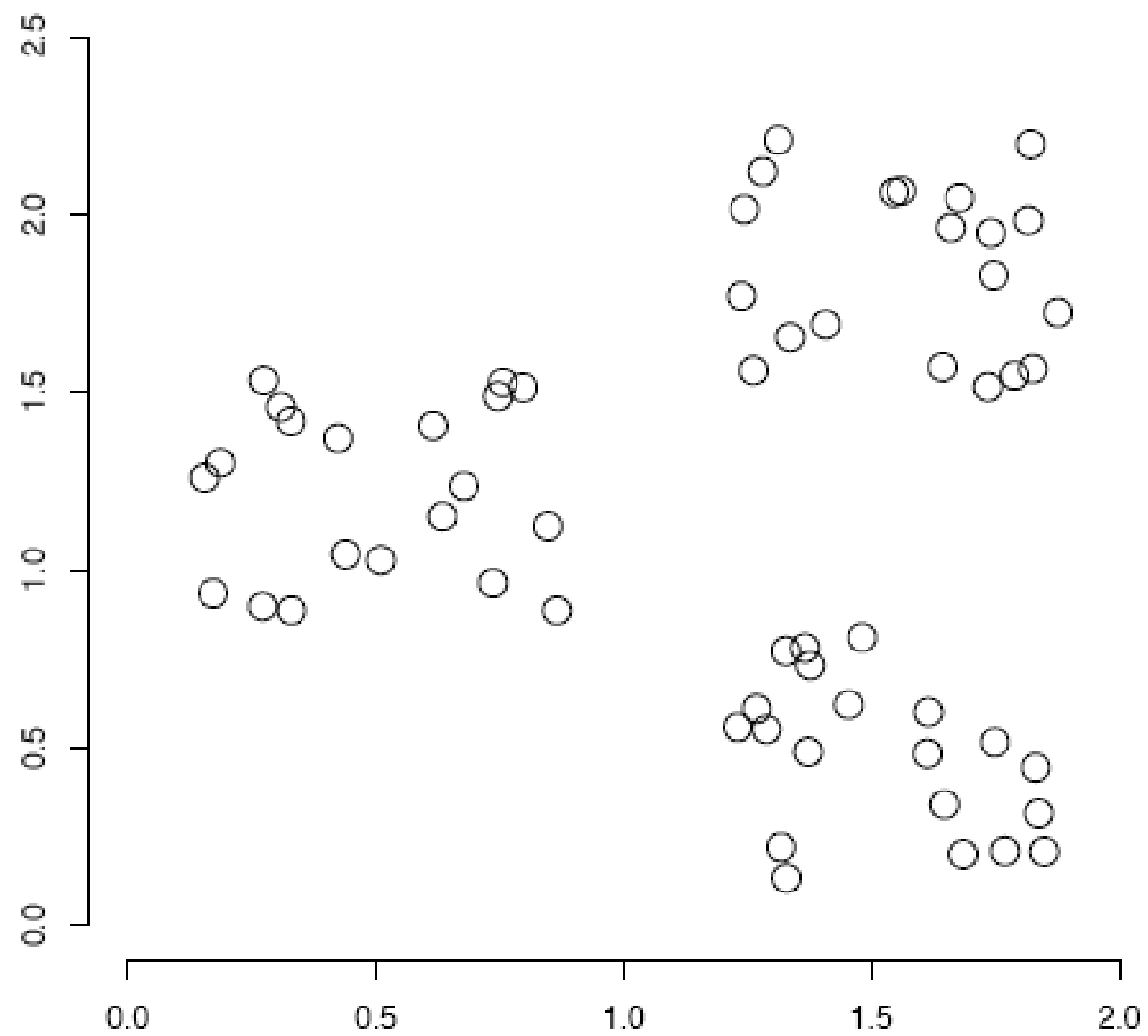
Hard vs. soft clustering

- Hard clustering: Each document belongs to exactly one cluster
 - More common and easier to do
- Soft clustering: A document can belong to more than one cluster.
 - Makes more sense for applications like creating browsable hierarchies

Flat algorithms

- Flat algorithms compute a partition of N documents into a set of K clusters.
- Given: a set of documents and the number K
- Find: a partition into K clusters that optimizes the chosen partitioning criterion
- Global optimization: exhaustively enumerate partitions, pick optimal one
 - Not tractable
- Effective heuristic method: K -means algorithm

K-means



K-means (in one slide!)

Input is **k** (the number of clusters), **data points** in Euclidean space

0. Initialize clusters by picking one point per cluster

Loop:

1. Place each point in the cluster whose current centroid is nearest
2. Find the new centroid for each cluster

K-means

- Objective/partitioning criterion: minimize the average squared difference from the centroid
- Assumes documents are real-valued vectors
- Clusters based on *centroids* (aka the *center of gravity* or mean) of points in a cluster, ω :

$$\vec{\mu}(\omega) = \frac{1}{|\omega|} \sum_{\vec{x} \in \omega} \vec{x}$$

- We try to find the minimum average squared difference by iterating two steps:
 - **reassignment**: assign each vector to its closest centroid
 - **recomputation**: recompute each centroid as the average of the vectors that were assigned to it in reassignment

K -MEANS($\{\vec{x}_1, \dots, \vec{x}_N\}, K$)

```
1   $(\vec{s}_1, \vec{s}_2, \dots, \vec{s}_K) \leftarrow \text{SELECTRANDOMSEEDS}(\{\vec{x}_1, \dots, \vec{x}_N\}, K)$ 
2  for  $k \leftarrow 1$  to  $K$ 
3  do  $\vec{\mu}_k \leftarrow \vec{s}_k$ 
4  while stopping criterion has not been met
5  do for  $k \leftarrow 1$  to  $K$ 
6      do  $\omega_k \leftarrow \{\}$ 
7      for  $n \leftarrow 1$  to  $N$ 
8          do  $j \leftarrow \arg \min_{j'} |\vec{\mu}_{j'} - \vec{x}_n|$ 
9               $\omega_j \leftarrow \omega_j \cup \{\vec{x}_n\}$  (reassignment of vectors)
10     for  $k \leftarrow 1$  to  $K$ 
11         do  $\vec{\mu}_k \leftarrow \frac{1}{|\omega_k|} \sum_{\vec{x} \in \omega_k} \vec{x}$  (recomputation of centroids)
12 return  $\{\vec{\mu}_1, \dots, \vec{\mu}_K\}$ 
```

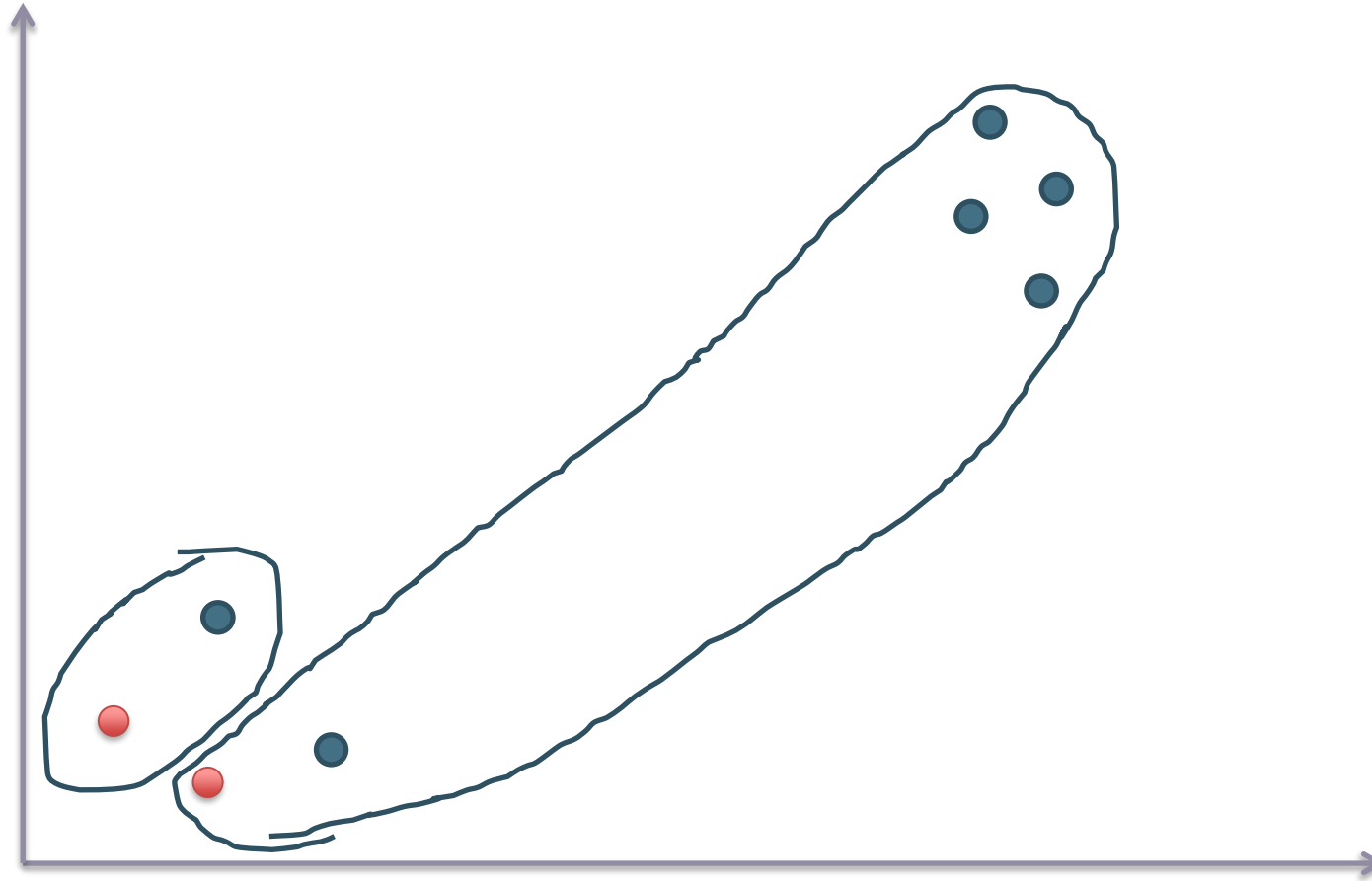
K-Means Clustering Example



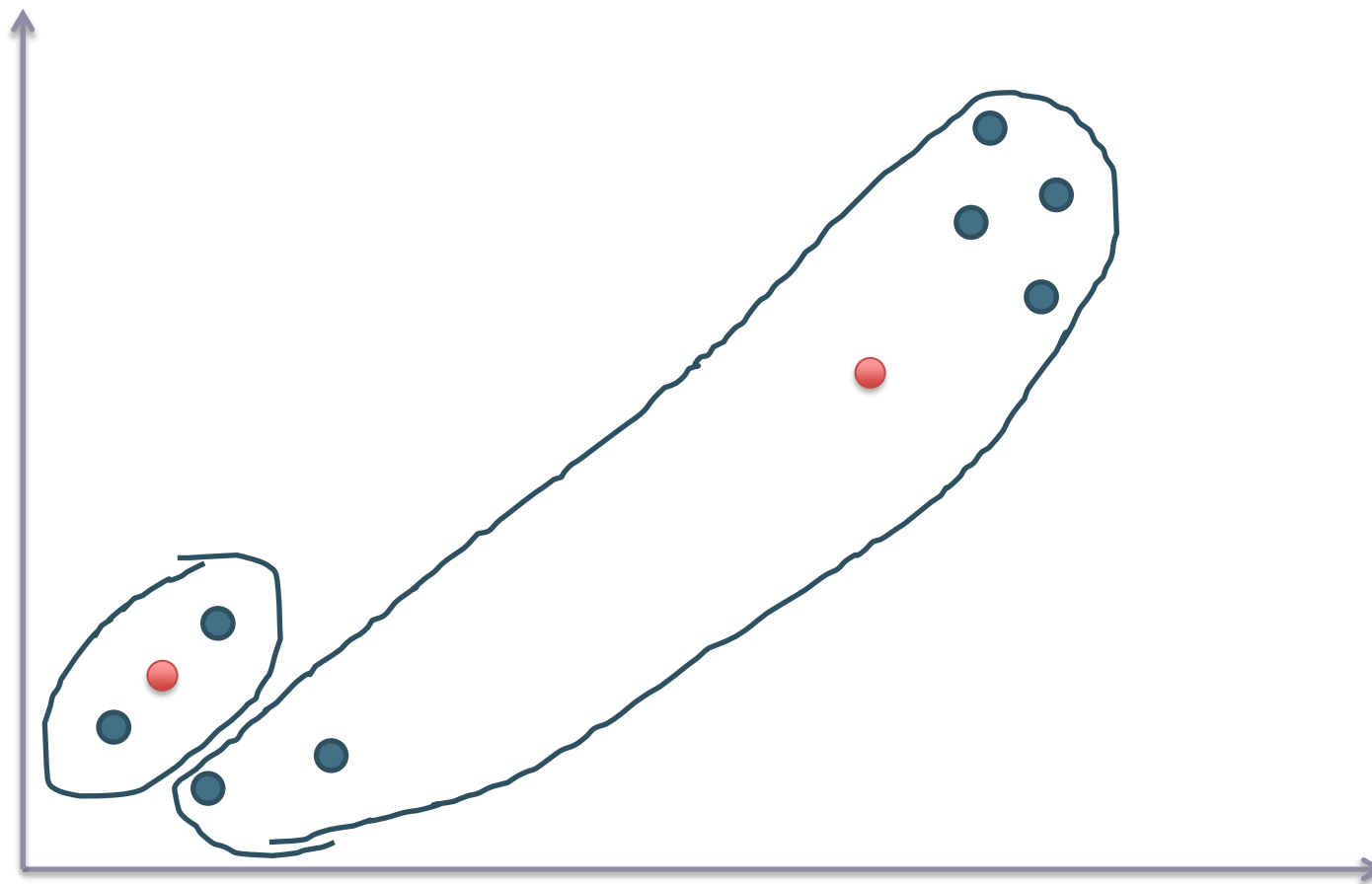
K-Means Clustering Example



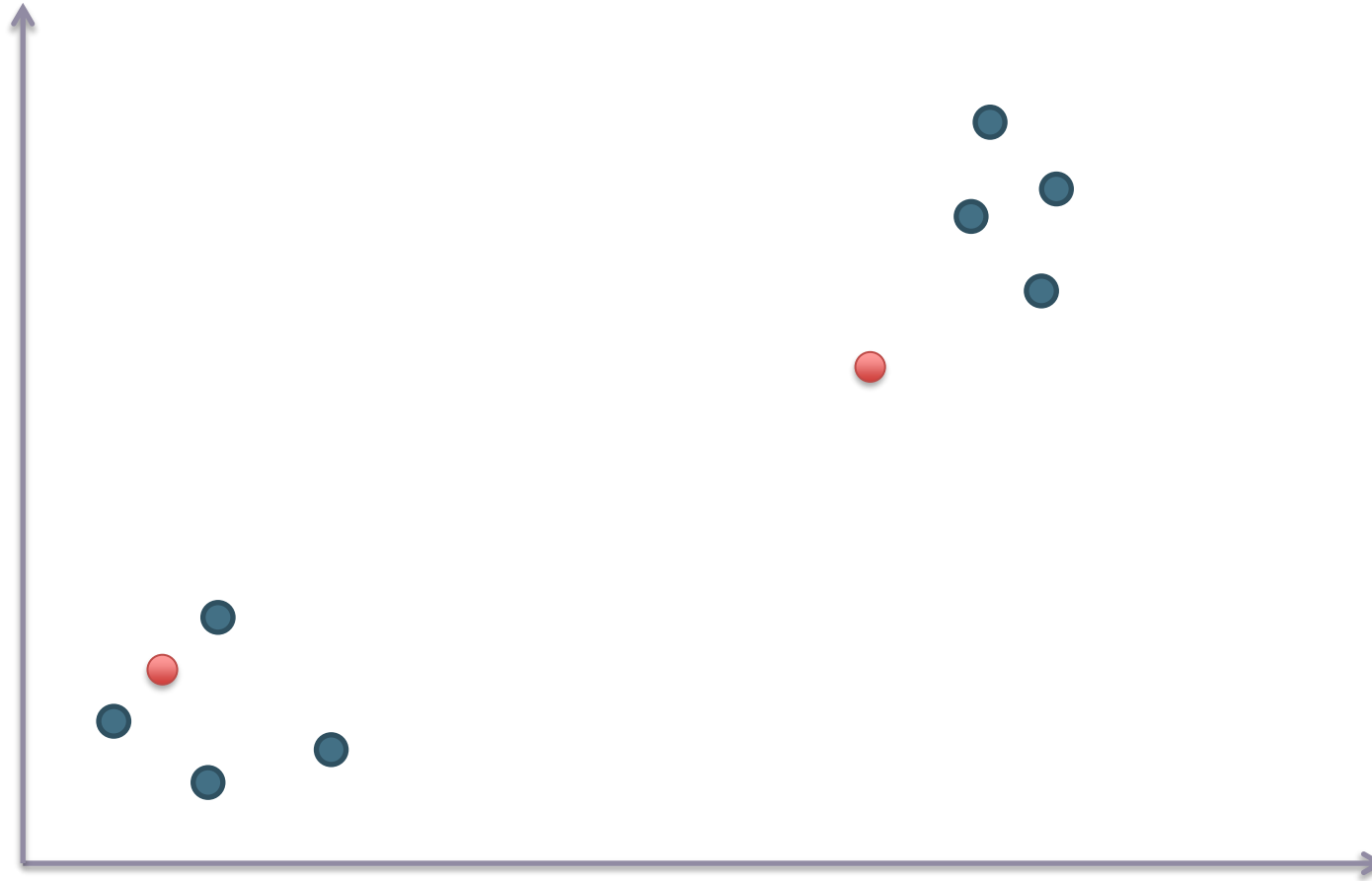
K-Means Clustering Example



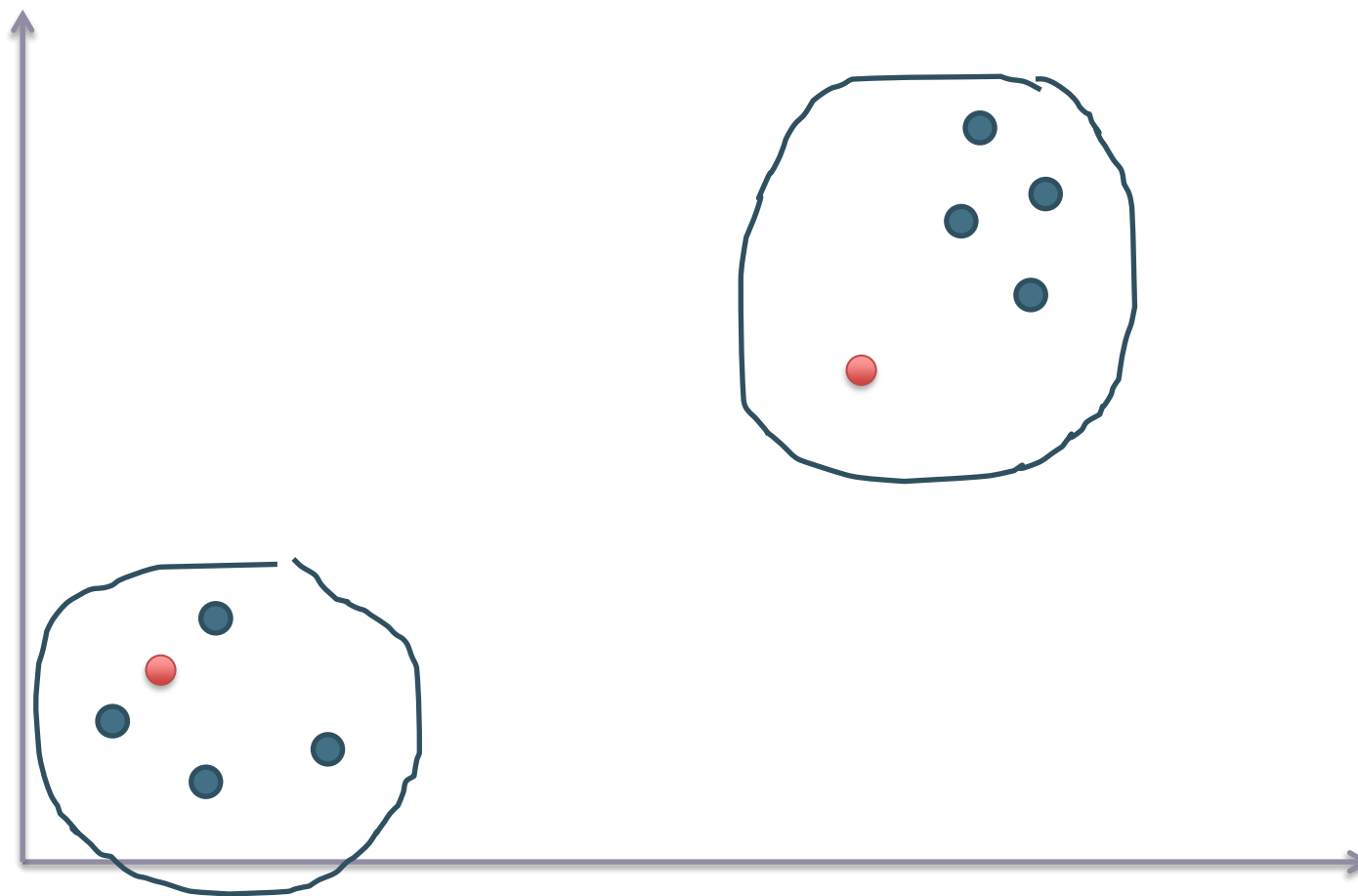
K-Means Clustering Example



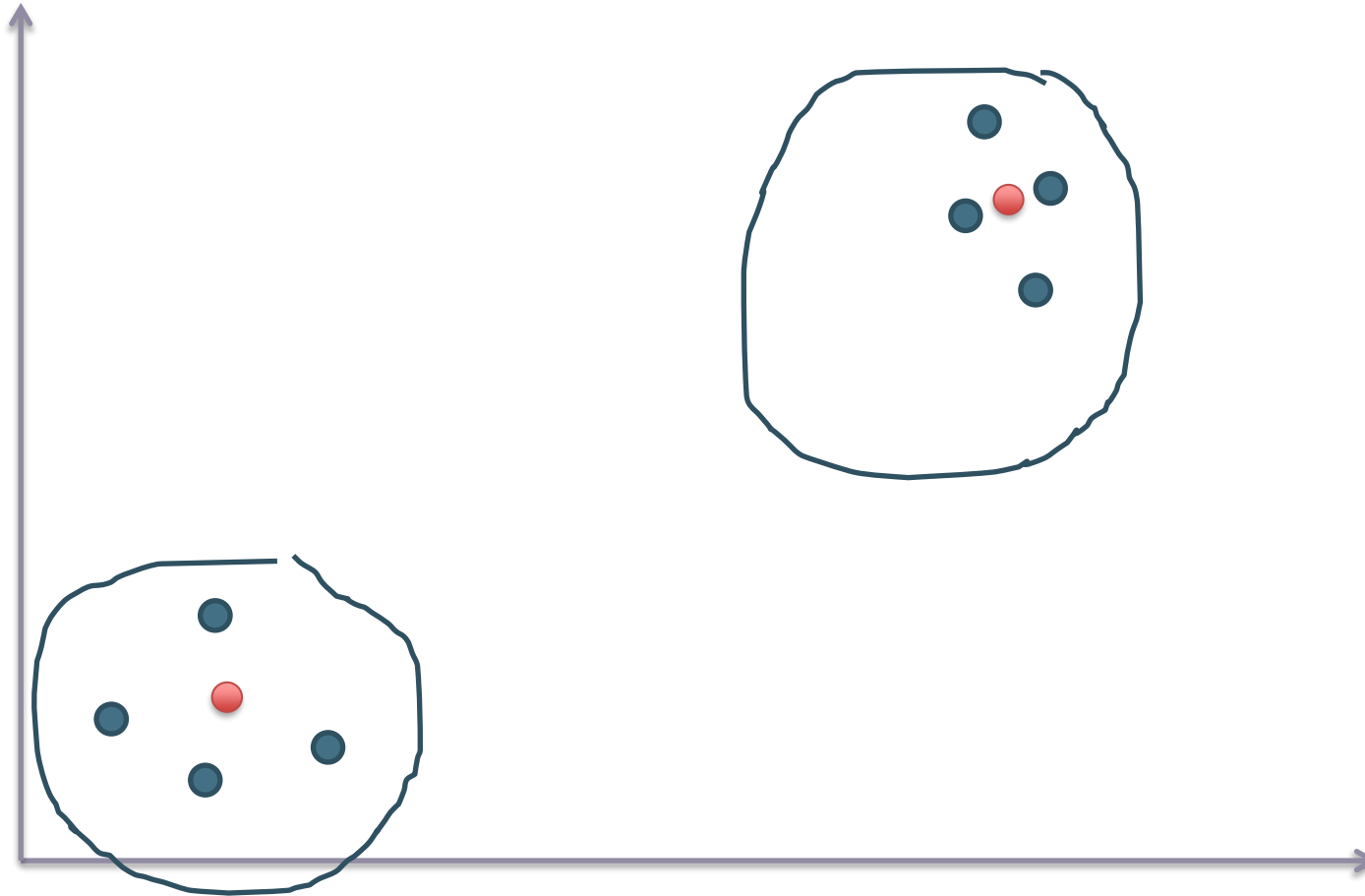
K-Means Clustering Example



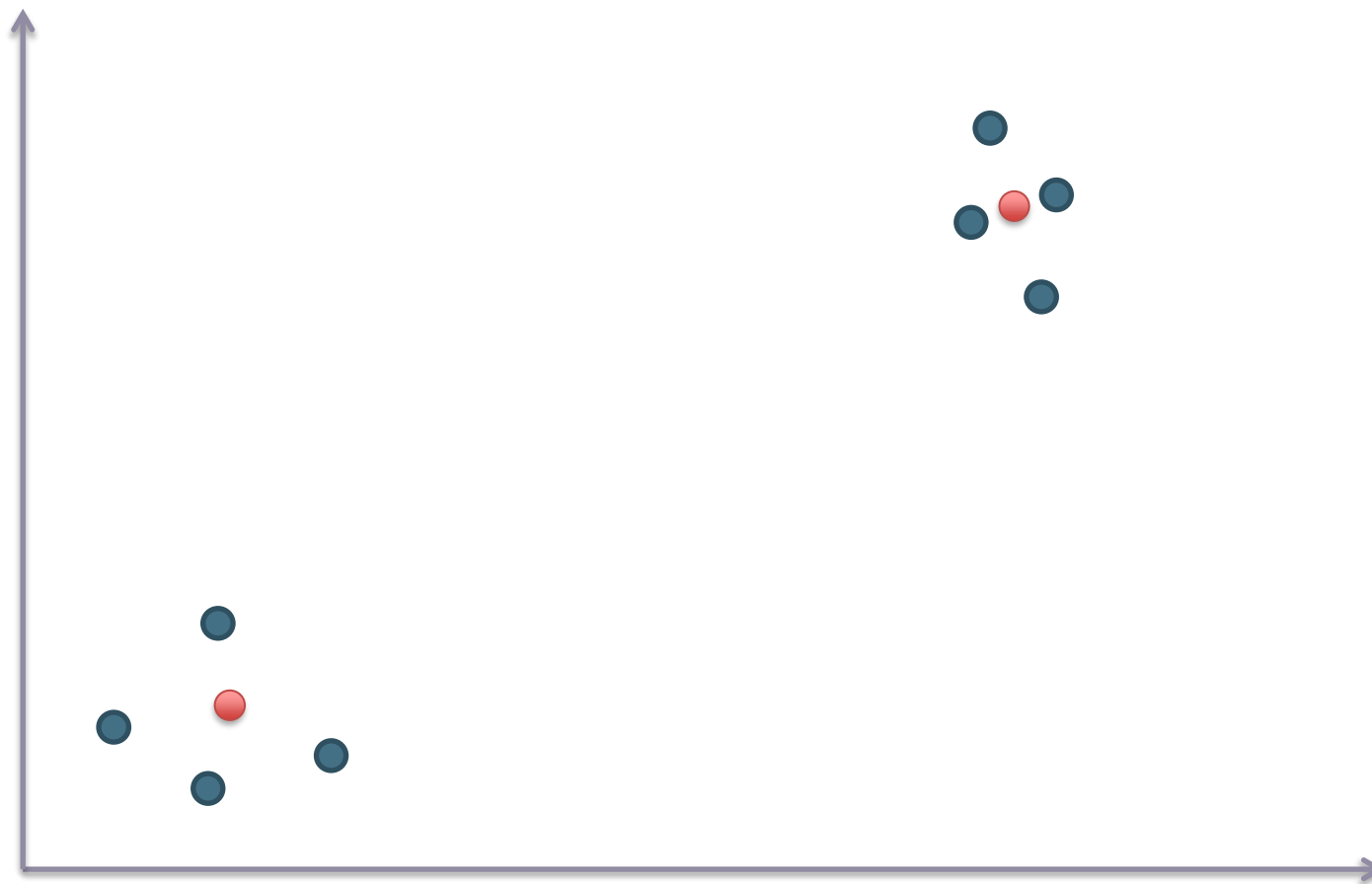
K-Means Clustering Example



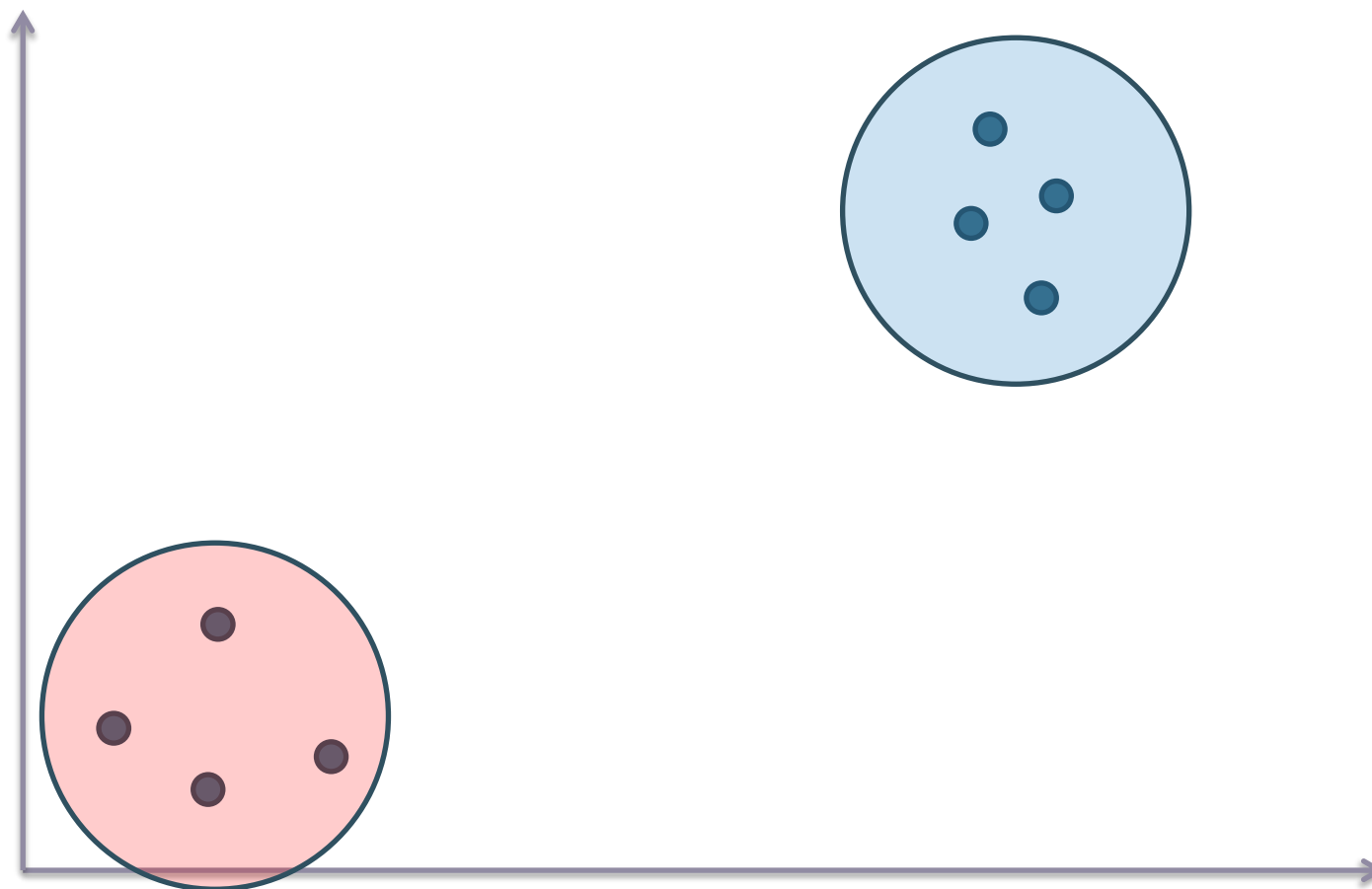
K-Means Clustering Example



K-Means Clustering Example

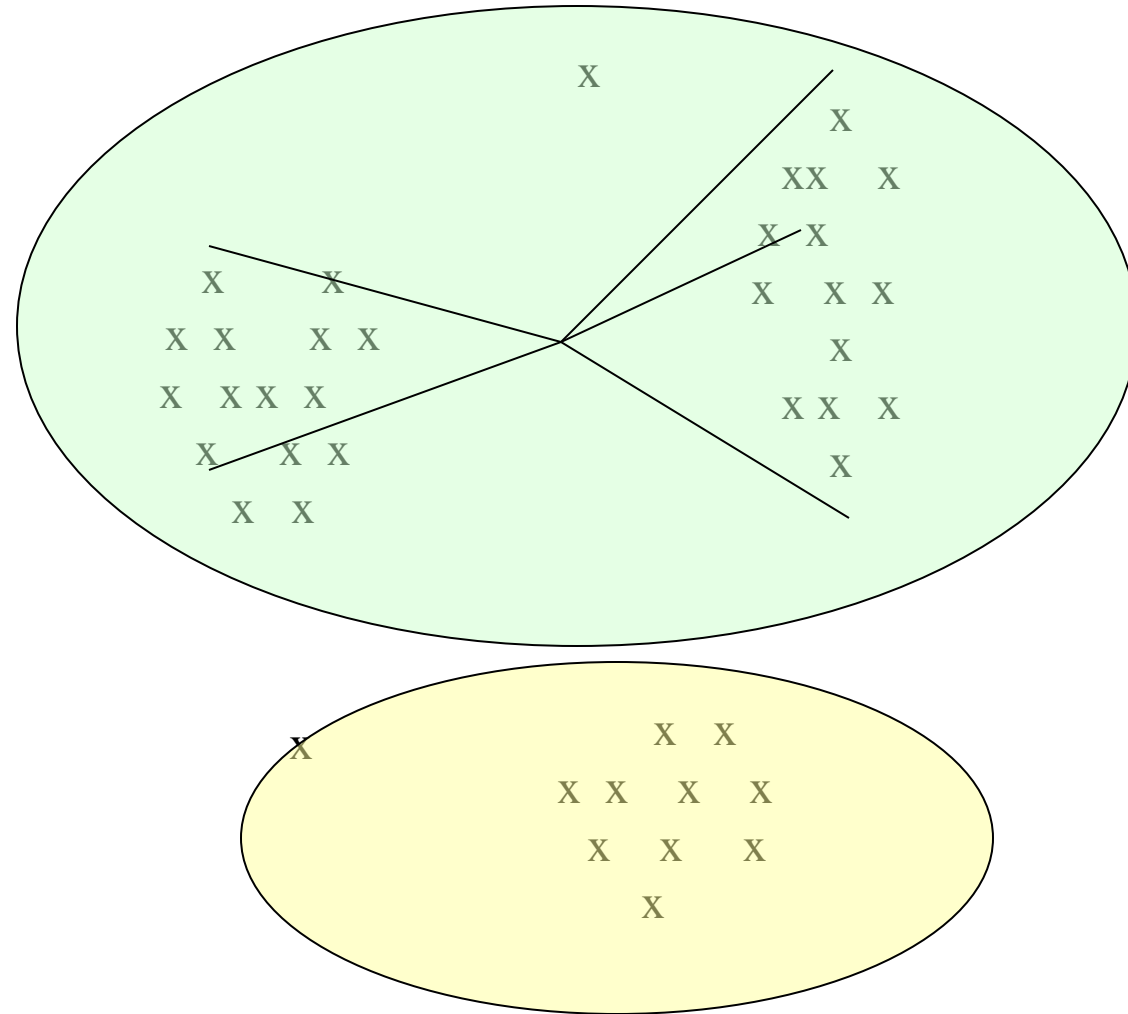


K-Means Clustering Example



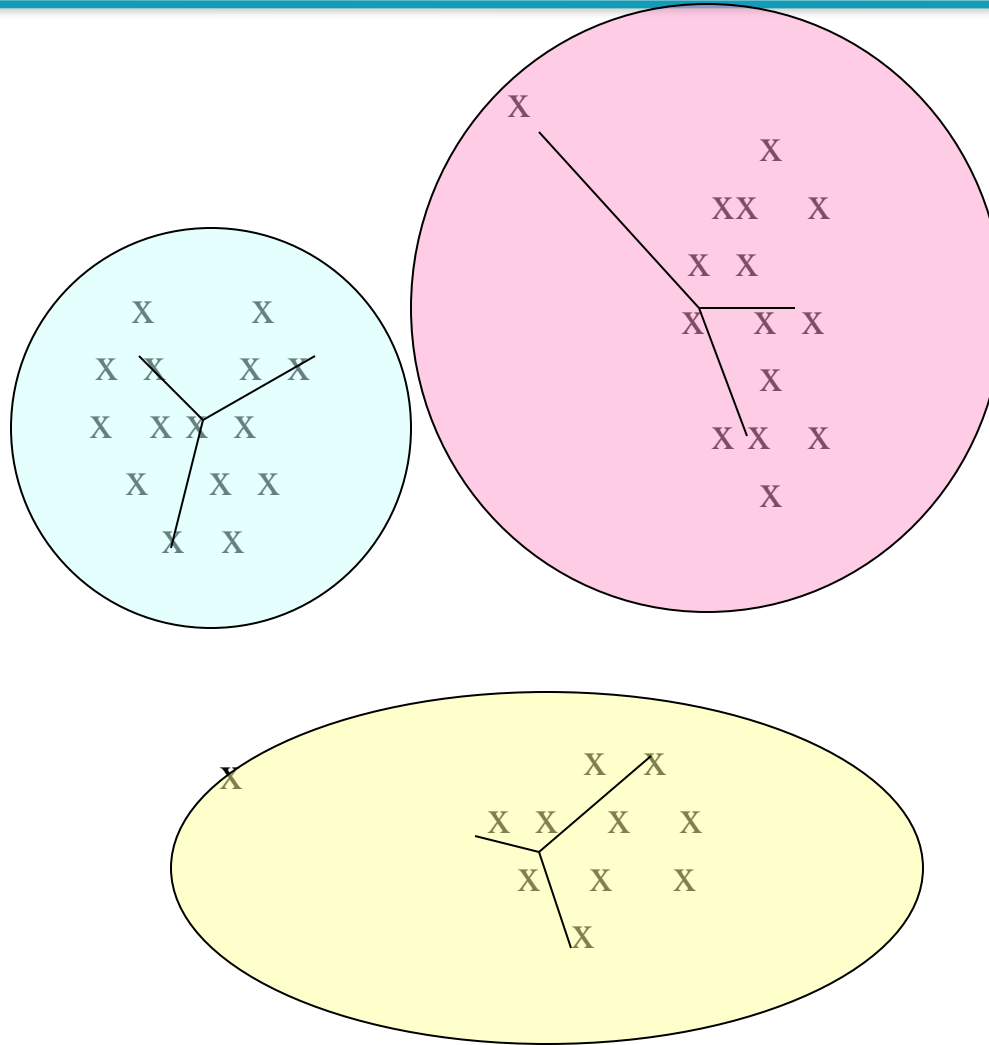
Example: Picking k

Too few;
many long
distances
to centroid.



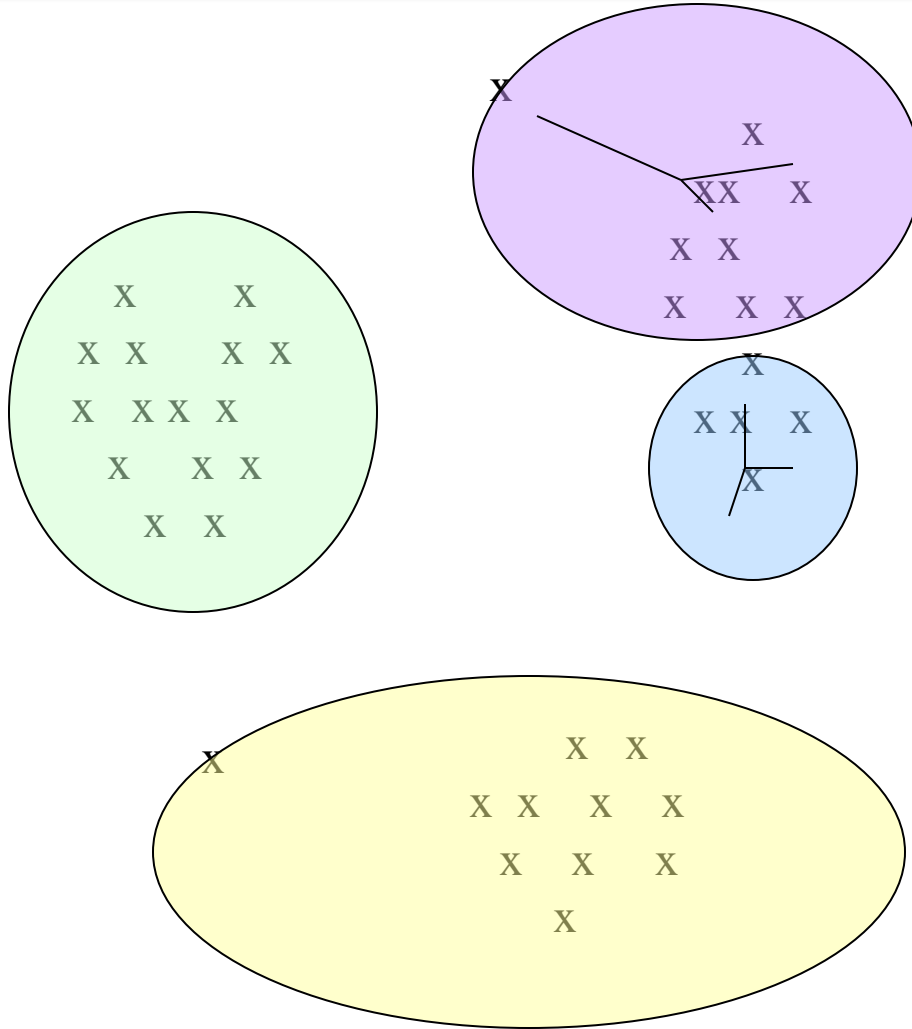
Example: Picking k

Just right;
distances
rather short.



Example: Picking k

Too many;
little improvement
in average
distance.



Convergence of K Means

- K-means converges to a fixed point in a finite number of iterations.
- Proof:
 - The sum of squared distances (RSS) decreases during reassignment.
 - (because each vector is moved to a closer centroid)
 - RSS decreases during recomputation.
 - There is only a finite number of clusterings
 - Thus: We must reach a fixed point.
- But we don't know how long convergence will take!
- If we don't care about a few docs switching back and forth, then convergence is usually fast (< 10-20 iterations).

Recomputation decreases average distance

- RSS = residual sum of squares (the “goodness” measure G)

$$\text{RSS}_k(\vec{v}) = \sum_{\vec{x} \in \omega_k} \|\vec{v} - \vec{x}\|^2 = \sum_{\vec{x} \in \omega_k} \sum_{m=1}^M (v_m - x_m)^2$$

$$\frac{\partial \text{RSS}_k(\vec{v})}{\partial v_m} = \sum_{\vec{x} \in \omega_k} 2(v_m - x_m) = 0$$

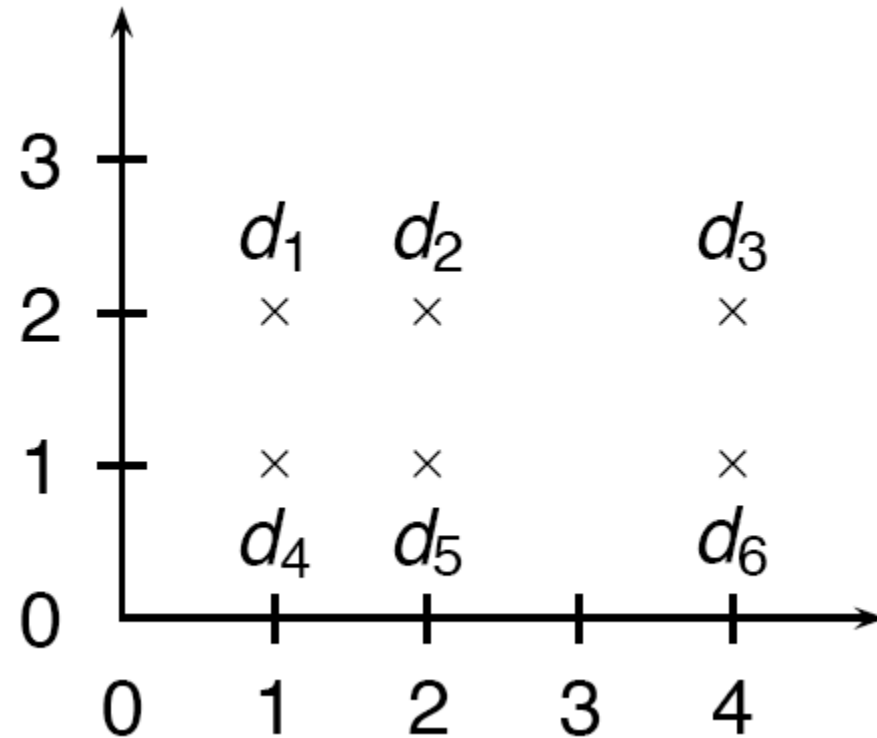
$$v_m = \frac{1}{|\omega_k|} \sum_{\vec{x} \in \omega_k} x_m$$

- The last line is the componentwise definition of the centroid! We minimize RSS_k when the old centroid is replaced with the new centroid. RSS, the sum of the RSS_k , must then also decrease during recomputation.

Optimality of K -means

- Convergence does not mean that we converge to the optimal clustering!
- This is the great weakness of K -means.
- If we start with a bad set of seeds, the resulting clustering can be horrible.

Example of suboptimal clustering!!!!



- What is the optimal clustering for $K=2$?
- What happens when our seeds are: d_2, d_5 ?

Initialization of K -means

- Results can vary based on random seed selection.
- Some seeds can result in poor convergence rate, or convergence to sub-optimal clusterings.
 - Select good seeds using a heuristic (e.g., doc least similar to any existing mean)
 - Try out multiple starting points
 - Initialize with the results of another method.

How many clusters?

Hmm...

- **Either: Number of clusters K is given.**
 - Then partition into K clusters
 - K might be given because there is some external constraint. Example: You cannot show more than 10–20 clusters on a screen.
- **Or: Finding the “right” number of clusters is part of the problem.**
 - Given docs, find K for which an optimum is reached.
 - How to define “optimum”?
 - Why can’t we use RSS or average distance from centroid?

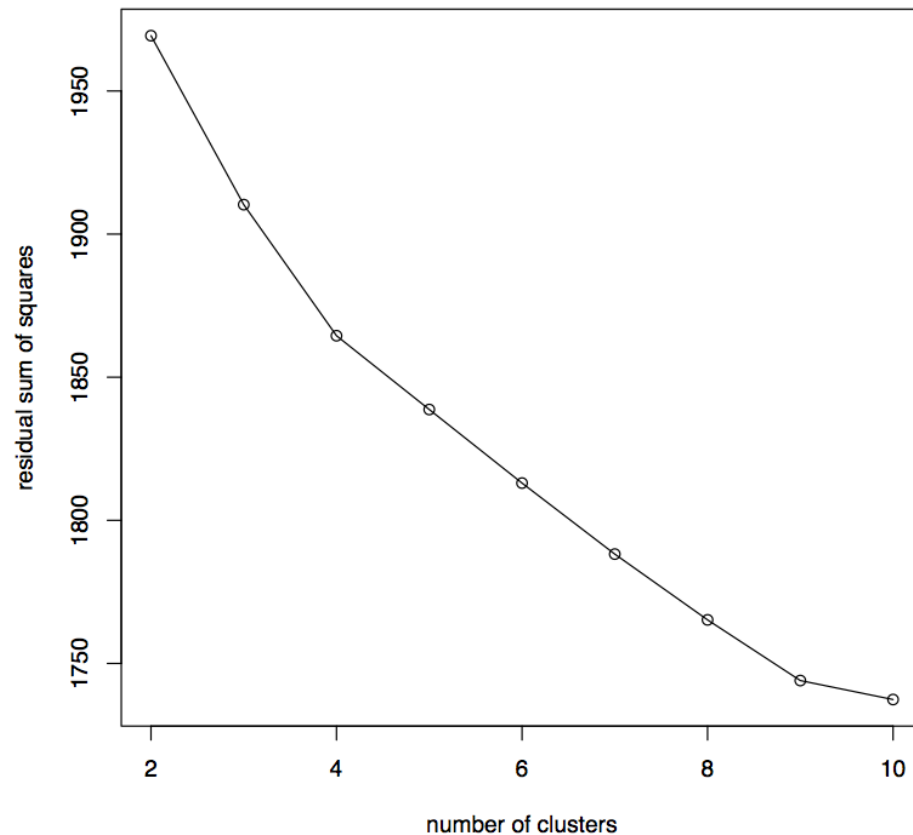
Simple objective function for K

- Basic idea:
 - Start with 1 cluster ($K = 1$)
 - Keep adding clusters (= keep increasing K)
 - Add a penalty for each new cluster
- Trade off cluster penalties against average squared distance from centroid
- Choose the value of K with the best tradeoff

Simple objective function for K

- Given a clustering, define the cost for a document as (squared) distance to centroid
- Define total **distortion** $RSS(K)$ as sum of all individual document costs (corresponds to average distance)
- Then: penalize each cluster with a cost λ
- Thus for a clustering with K clusters, total cluster penalty is $K\lambda$
- Define the total cost of a clustering as distortion plus total cluster penalty: $RSS(K) + K\lambda$
- Select K that minimizes $(RSS(K) + K\lambda)$
- Still need to determine good value for λ . . .

Finding the “knee” in the curve



Pick the number of clusters where curve “flattens”. Here: 4 or 9.

Labeling

Major issue - labeling

- After clustering algorithm finds clusters - how can they be useful to the end user?
- Need simple label for each cluster
 - In search results, say “Animal” or “Car” in the *jaguar* example.
 - In topic trees (Yahoo), need navigational cues.
 - Often done by hand, a posteriori.

Ideas?

- Use metadata like Titles
- Use the medoid (document) itself – Title
- Top-terms (most frequent)
 - Stop-words, duplicates
- Most distinguishing terms (in my cluster, not in your cluster)
 - Measure Mutual Information

How to Label Clusters

- Show titles of typical documents
 - Titles are easy to scan
 - Authors create them for quick scanning!
 - But you can only show a few titles which may not fully represent cluster
- Show words/phrases prominent in cluster
 - More likely to fully represent cluster
 - Use distinguishing words/phrases
 - Differential labeling
 - But harder to scan

Labeling

- Common heuristics - list 5-10 most frequent terms in the centroid vector.
 - Drop stop-words; stem.
- Differential labeling by frequent terms
 - Within a collection “Computers”, clusters all have the word ***computer*** as frequent term.
 - Discriminant analysis of centroids.
- Perhaps better: distinctive noun phrase

Cluster labeling: example

	# docs	labeling method		
		centroid	mutual information	title
4	622	oil plant mexico production crude power 000 refinery gas bpd	plant oil production barrels crude bpd mexico dolly capa- city petroleum	MEXICO: Hurricane Dolly heads for Me- xico coast
9	1017	police security rus- sian people milita- ry peace killed told grozny court	police killed military security peace told troops forces re- bels people	RUSSIA: Russia's Lebed meets rebel chief in Chechnya
10	1259	00 000 tonnes tra- ders futures wheat prices cents sep- tember tonne	delivery traders fu- tures tonne tonnes desk wheat prices 000 00	USA: Export Business - Grain/oilseeds complex

- Three methods: most prominent terms in centroid, differential labeling using MI, title of doc closest to centroid
- Any feature selection method can also be used for labeling

Final word

- In clustering, clusters are inferred from the data without human input (unsupervised learning)
- However, in practice, it's a bit less clear: there are many ways of influencing the outcome of clustering: number of clusters, similarity measure, representation of documents, . . .

Evaluation

What is a good clustering?

- **Internal criteria**
 - Example of an internal criterion: RSS in K-means
- But an internal criterion often does not evaluate the actual utility of a clustering in the application
- **Alternative: External criteria**
 - Evaluate with respect to human-defined classification
 - Require the ground truth

External criteria for clustering quality

- Based on a gold standard data set, e.g., the Reuters collection
- Goal: Clustering should reproduce the classes in the gold standard
- (But we only want to reproduce how documents are divided into groups, not the class labels.)
- First measure for how well we were able to reproduce the classes:
purity

External criterion: Purity

$$\text{purity}(\Omega, \Gamma) = \frac{1}{N} \sum_k \max_j |\omega_k \cap c_j|$$

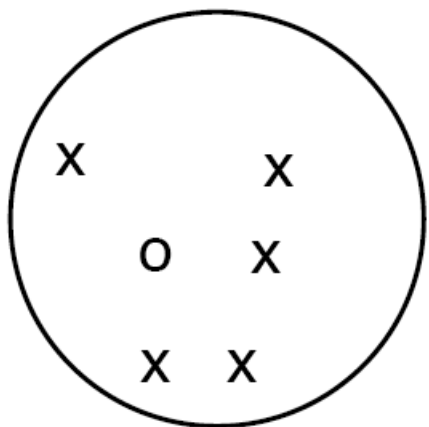
$\Omega = \{\omega_1, \omega_2, \dots, \omega_K\}$ is the set of clusters and
 $\Gamma = \{c_1, c_2, \dots, c_J\}$ is the set of classes.

For each cluster ω_k : find class c_j with most members n_{kj} in cluster

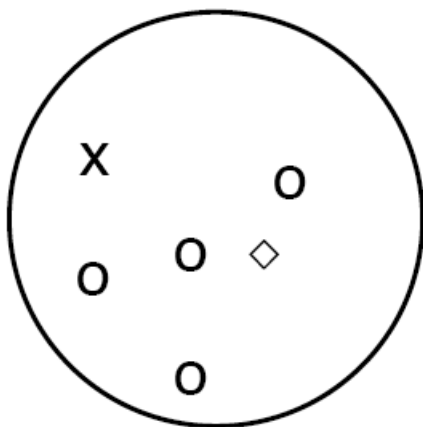
Sum all n_{kj} and divide by total number of points

Example

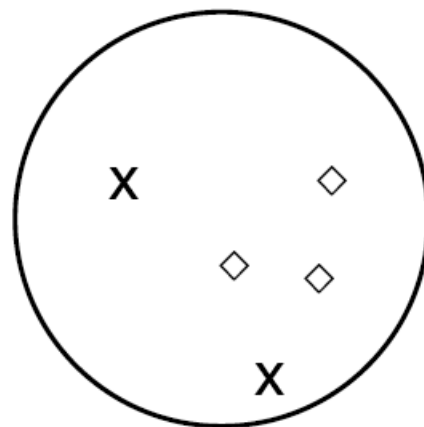
cluster ω_1



cluster ω_2



cluster ω_3



$$\text{good_docs}(\omega_1) = \max(5, 1, 0) = 5$$

$$\text{good_docs}(\omega_2) = \max(1, 4, 1) = 4$$

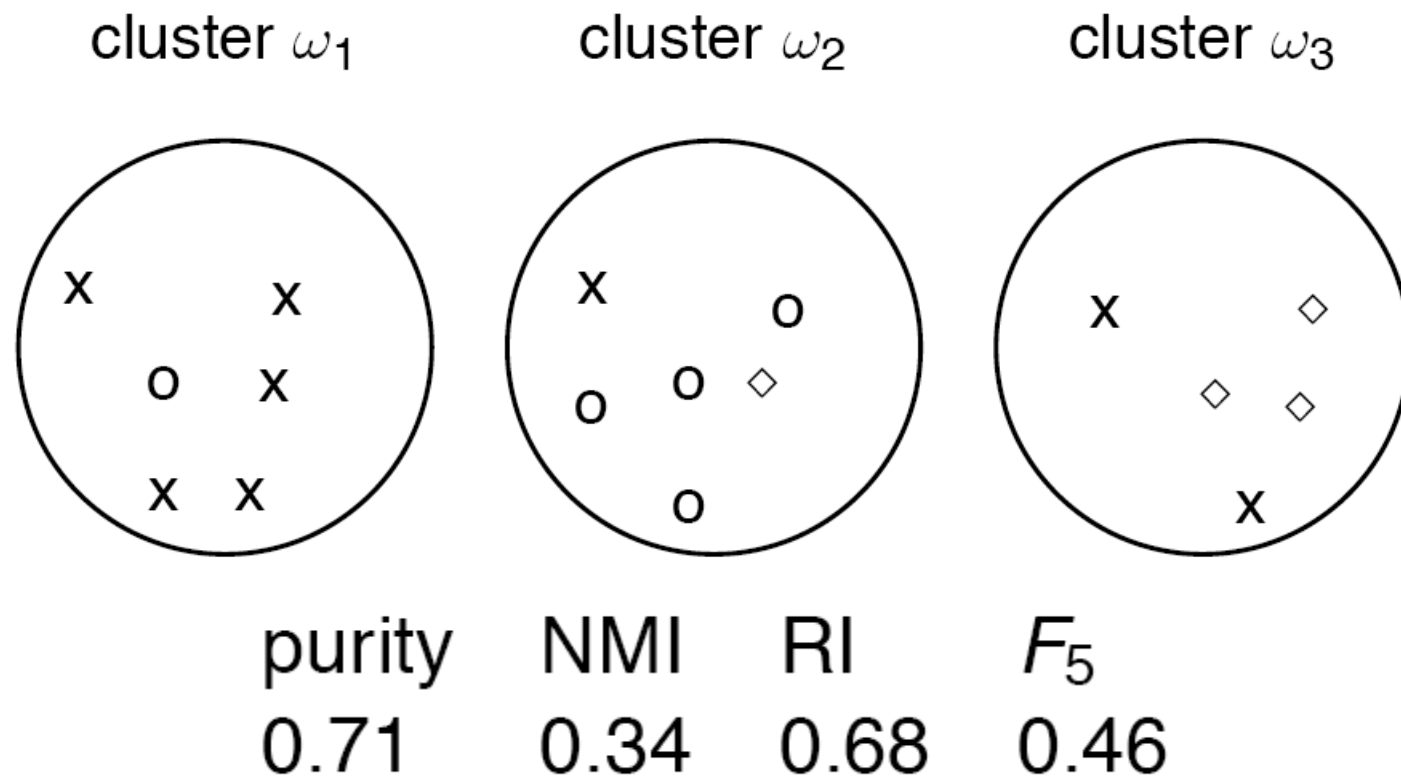
$$\text{good_docs}(\omega_3) = \max(2, 0, 3) = 3$$

$$\text{purity}(\Omega) = 1/17 \cdot (5 + 4 + 3) = 12/17$$

Three other external evaluation measures

- Rand Index
- Normalized mutual information (NMI)
 - How much information does the clustering contain about the classification?
 - Singleton clusters (number of clusters = number of docs) have maximum MI
 - Therefore: normalize by entropy of clusters and classes
- F measure
 - Like Rand, but “precision” and “recall” can be weighted

Evaluation results



- All four measures range from 0 (really bad clustering) to 1 (perfect clustering).