

# Machine Learning for IR

---

Claudia Hauff

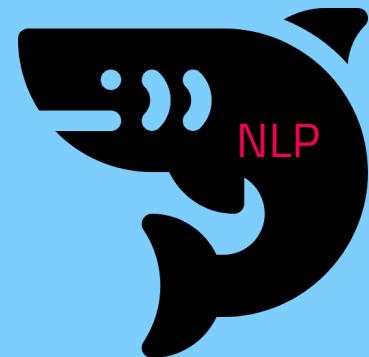
ESSIR 2019, July 16, 2019.

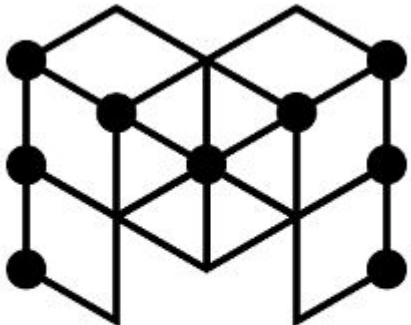


Machine learning  
adapted to IR problems

# Learning to Rank

Neural IR





## MS MARCO

<http://www.msmarco.org>

Given a query (1M total) and a corpus of 9M passages, rank the passages by relevance. Use either the full corpus or start with BM25's top 1K passages.

symptoms of an enlarged heart in dogs  
how long can i freeze pork  
lewisville texas is in what county  
...

## Passage Retrieval(10/26/2018-Present)

Rank	Model	Ranking Style	Submission Date	MRR@10 On Eval
1	Enriched BERT base + AOA index V1 Ming Yan of Alibaba Damo NLP	Full Ranking	May 13th, 2019	0.383
2	BERTter pretraining (1)Rodrigo Nogueira, (2)Wei Yang, (3)Jimmy Lin, (4)Kyunghyun Cho - New York University(1,4), University of Waterloo(2,3), Facebook AI Research(4)	Full Ranking	May 21st, 2019	0.383
3	Enriched BERT base + AOA index V2 Ming Yan of Alibaba Damo NLP	Full Ranking	May 13th, 2019	0.380
4	BM25 + monoBERT + duoBERT + TCP Anonymous	Full Ranking	June 26th, 2019	0.379
5	BERT^2 (1)Rodrigo Nogueira, (2)Wei Yang, (3)Jimmy Lin, (4)Kyunghyun Cho - New York University(1,4), University of Waterloo(2,3), Facebook AI Research(4)	Full Ranking	May 13th, 2019	0.375
6	Enriched BERT base + AOA index Ming Yan of Alibaba Damo NLP	Full Ranking	May 6th, 2019	0.373
7	BM25 + monoBERT + duoBERT Anonymous	Full Ranking	June 26th, 2019	0.370
8	BERTter Indexing (1)Rodrigo Nogueira, (2)Wei Yang, (3)Jimmy Lin, (4)Kyunghyun Cho - New York University(1,4), University of Waterloo(2,3), Facebook AI Research(4) [Nogueira et al. '19] and [Code]	Full Ranking	April 8th, 2019	0.368
9	Enriched BERT base + AOA index Ming Yan of Alibaba Damo NLP	ReRanking	May 6th, 2019	0.368
10	BM25 + monoBERT Anonymous	Full Ranking	June 26th, 2019	0.365
11	SAN + BERT base Yu Wang, Xiaodong Liu, Jianfeng Gao - Deep Learning Group, Microsoft Research AI [Xiaodong, et al. '18]	Ref king	January 22th, 2019	0.359
12	BERT + Small Training Rodrigo Nogueira(1) and Kyunghyun Cho(2) - New York University(1,2), Facebook AI Research(2) [Nogueira, et al. '19]	Ref king	January 7th, 2019	0.359

Given a query (1M total) and a corpus of 9M passages, rank the passages by relevance. Use either the full corpus or start with BM25's top 1K passages.

symptoms of an enlarged heart in dogs  
how long can i freeze pork  
lewisville texas is in what county

...

40	Feature-based LeToR: simple-feature based RankSVM(1)Yifan Qiao, (2)Chenyan Xiong, (3)Zhenghao Liu, (4)Zhiyuan Liu-Tsinghua University(1, 3, 4); Microsoft Research AI(2)	ReRanking	December 10th, 2018	0.191
41	BM25 (Lucene8, tuned) Anonymous	Full Ranking	June 26th, 2019	0.190
42	BM25 (Anserini) (1)Rodrigo Nogueira, (2)Wei Yang, (3)Jimmy Lin, (4)Kyunghyun Cho - New York University(1,4), University of Waterloo(2,3), Facebook AI Research(4)[Nogueira et al. '19] and [Code]	Full Ranking	April 10th, 2019	0.186
43	Unnamed Hongyin Zhu	Ref king	June 26th, 2019	0.174
44	[Official Baseline]BM25 Stephen E. Robertson; Steve Walker; Susan Jones; Micheline Hancock-Beaulieu & Mike Gatford (Implemented by MSMARCO Team) [ Robertson et al. '94]	Full Ranking	Novmeber 1st, 2018	0.165

## Passage Retrieval(10/26/2018-Present)

Rank	Model	Ranking Style	Submission Date	
1	Enriched BERT base + AOA index V1 Ming Yan of Alibaba Damo NLP	Full Ranking	May 13th, 2019	0.383
2	BERTter pretraining (1)Rodrigo Nogueira, (2)Wei Yang, (3)Jimmy Lin, (4)Kyunghyun Cho - New York University(1,4), University of Waterloo(2,3), Facebook AI Research(4)	Full Ranking	May 21st, 2019	0.383
3	Enriched BERT base + AOA index V2 Ming Yan of Alibaba Damo NLP	Full Ranking	May 13th, 2019	0.380
4	BM25 + monoBERT + duoBERT + TCP Anonymous	Full Ranking	June 26th, 2019	0.379
5	BERT^2 (1)Rodrigo Nogueira, (2)Wei Yang, (3)Jimmy Lin, (4)Kyunghyun Cho - New York University(1,4), University of Waterloo(2,3), Facebook AI Research(4)	Full Ranking	May 13th, 2019	0.375
6	Enriched BERT base + AOA index Ming Yan of Alibaba Damo NLP	Full Ranking	May 6th, 2019	0.373
7	BM25 + monoBERT + duoBERT Anonymous	Full Ranking	June 26th, 2019	0.370
8	BERTter Indexing (1)Rodrigo Nogueira, (2)Wei Yang, (3)Jimmy Lin, (4)Kyunghyun Cho - New York University(1,4), University of Waterloo(2,3), Facebook AI Research(4) [Nogueira et al. '19] and [Code]	Full Ranking	April 8th, 2019	0.368
9	Enriched BERT base + AOA index Ming Yan of Alibaba Damo NLP	ReRanking	May 6th, 2019	0.368
10	BM25 + monoBERT Anonymous	Full Ranking	June 26th, 2019	0.365
11	SAN + BERT base Yu Wang, Xiaodong Liu, Jianfeng Gao - Deep Learning Group, Microsoft Research AI [Xiaodong, et al. '18]	Ref king	January 22th, 2019	0.359
12	BERT + Small Training Rodrigo Nogueira(1) and Kyunghyun Cho(2) - New York University(1,2), Facebook AI Research(2) [Nogueira, et al. '19]	Ref king	January 7th, 2019	0.359

Given a query (1M total) and a corpus of 9M passages, rank the passages by relevance. Use either the full corpus or start with BM25's top 1K passages.

symptoms of an enlarged heart in dogs  
how long can i freeze pork  
lewisville texas is in what county

...

40	Feature-based LeToR: simple-feature based RankSVM(1)Yifan Qiao, (2)Chenyan Xiong, (3)Zhenghao Liu, (4)Zhiyuan Liu-Tsinghua University(1, 3, 4); Microsoft Research AI(2)	ReRanking	Decem 10th, 2
41	BM25 (Lucene8, tuned) Anonymous	Full Ranking	June 26th, 2019
42	BM25 (Anserini) (1)Rodrigo Nogueira, (2)Wei Yang, (3)Jimmy Lin, (4)Kyunghyun Cho - New York University(1,4), University of Waterloo(2,3), Facebook AI Research(4)[Nogueira et al. '19] and [Code]	Full Ranking	April 10th, 2019
43	Unnamed Hongyin Zhu	Ref king	June 26th, 2019
44	[Official Baseline]BM25 Stephen E. Robertson; Steve Walker; Susan Jones; Micheline Hancock-Beaulieu & Mike Gatford (Implemented by MSMARCO Team) [ Robertson et al. '94]	Full Ranking	Novmeber 1st, 2018



## BERT: Pre-training of Deep Bidirectional Transformers for Language ...

<https://arxiv.org> › cs ▾



by J Devlin - 2018 - Cited by 776 - Related articles

Oct 11, 2018 - Unlike recent language representation models, **BERT** is designed to ... As a result, the pre-trained **BERT** model can be fine-tuned with just one ... (or arXiv:1810.04805v2 [cs. ... Which authors of this paper are endorsers?

## XLNet: Generalized Autoregressive Pretraining for Language ...

<https://arxiv.org> › cs ▾

by Z Yang - 2019 - Cited by 1 - Related articles

Jun 19, 2019 - arXiv.org > cs > arXiv:1906.08237 ... In light of these pros and cons, we propose **XLNet**, a generalized ... Furthermore, **XLNet** integrates ideas from Transformer-XL, the state-of-the-art autoregressive model, into pretraining.

Yang, (3)Jimmy Lin, (4)Kyunghyun Cho - New York University(1,4), University of Waterloo(2,3), Facebook AI Research(4) [Nogueira et al. '19] and [Code]

Ranking 2019

9 Enriched BERT base + AOA index Ming Yan of Alibaba Damo NLP

ReRanking May 6th, 2019

10 BM25 + monoBERT Anonymous

Full Ranking June 26th, 2019

11 SAN + BERT base Yu Wang, Xiaodong Liu, Jianfeng Gao - Deep Learning Group, Microsoft Research AI [Xiaodong, et al. '18]

ReRanking January 22th, 2019

12 BERT + Small Training Rodrigo Nogueira(1) and Kyunghyun Cho(2) - New York University(1,2), Facebook AI Research(2) [Nogueira, et al. '19]

ReRanking January 7th 2019



Kyunghyun Cho  
@kchonyc

Following

re "Why not consider other models? such as XLNet": I agree with the reviewer on the importance of time travel research, but it's slightly out of the scope of this paper.

1:44 AM - 13 Jul 2019

54 Retweets 317 Likes



10

54

317

✉

# Learning to Rank

[1] Liu, Tie-Yan. "Learning to rank for information retrieval." *Foundations and Trends® in Information Retrieval* Vol. 3, No. 3 (2009): 225-331.

# Conventional ranking models in IR

## Query-dependent models

Vector space model

Boolean model

BM25

Language modeling

...

## Query-independent models

PageRank

TrustRank

Spaminess

Readability

...

How can we **combine** a  
large number of models  
to obtain an even better  
model?

# Overview

## Learning-to-rank

in the broad sense are all methods that use machine learning to solve the problem of ranking

(e.g. relevance feedback, hyperparameter tuning of BM25 ...)

## Learning-to-rank

in the narrow sense are all methods that learn the optimal way to combine **features** extracted from query-document pairs through **discriminative** training

Learns the conditional probability distribution  $P(y|x)$ . Generative training learns  $P(x,y)$  instead.

Given your machine learning background, how would you go about using machine learning to **learn a ranking function?**

*Ignore deep learning for now ...*

What is the input?

What is the output?

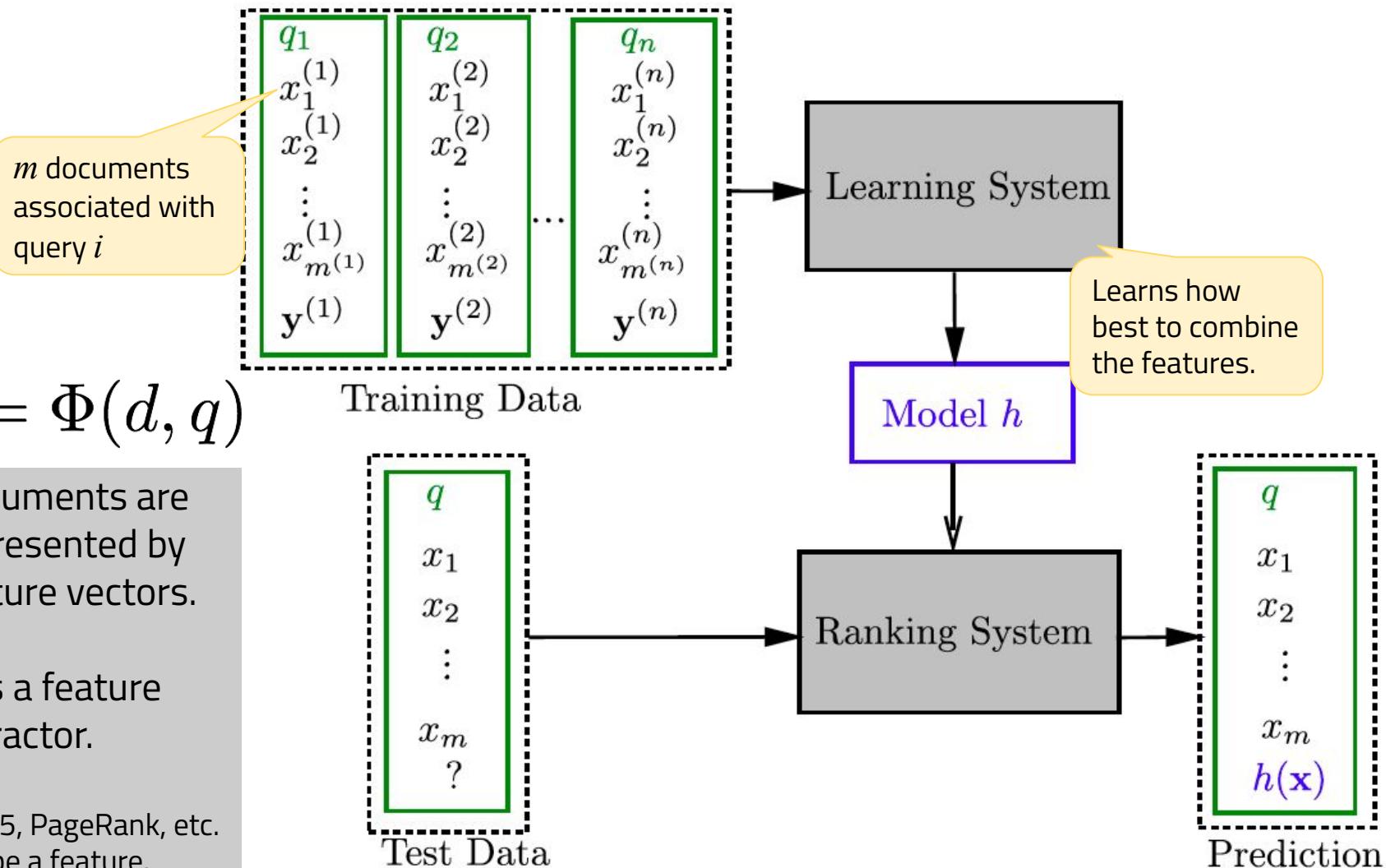
What is your success metric?





Supervised or  
unsupervised

# L2R setup



# Document judgment strategies

with respect to a query (topic)

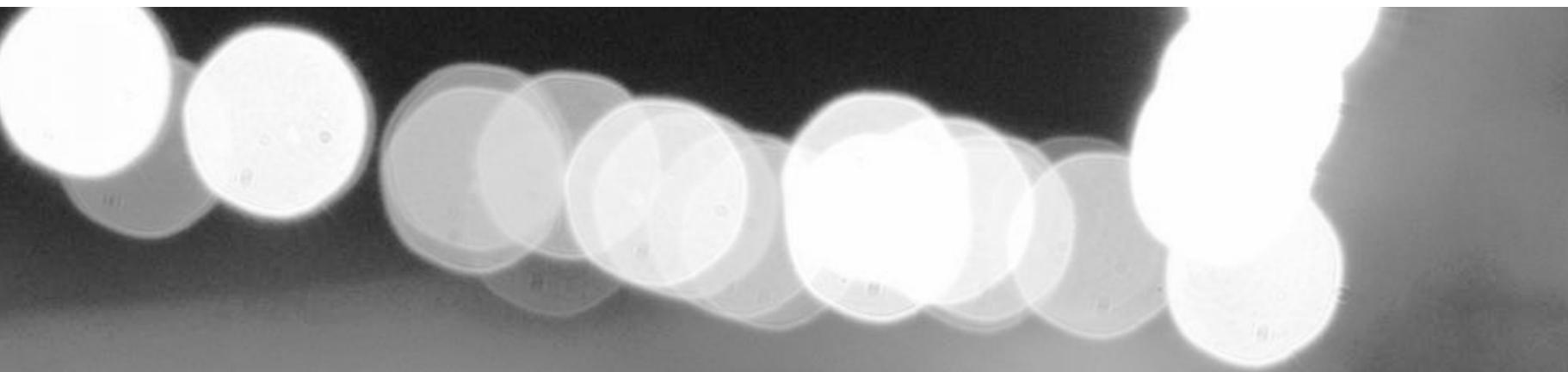
- Specifying whether a document is relevant (binary) or specifying a degree of relevance
- Specifying whether a document is *more* relevant than another one (relative preference)
- Specifying the partial or total order of documents (a set of permutations)

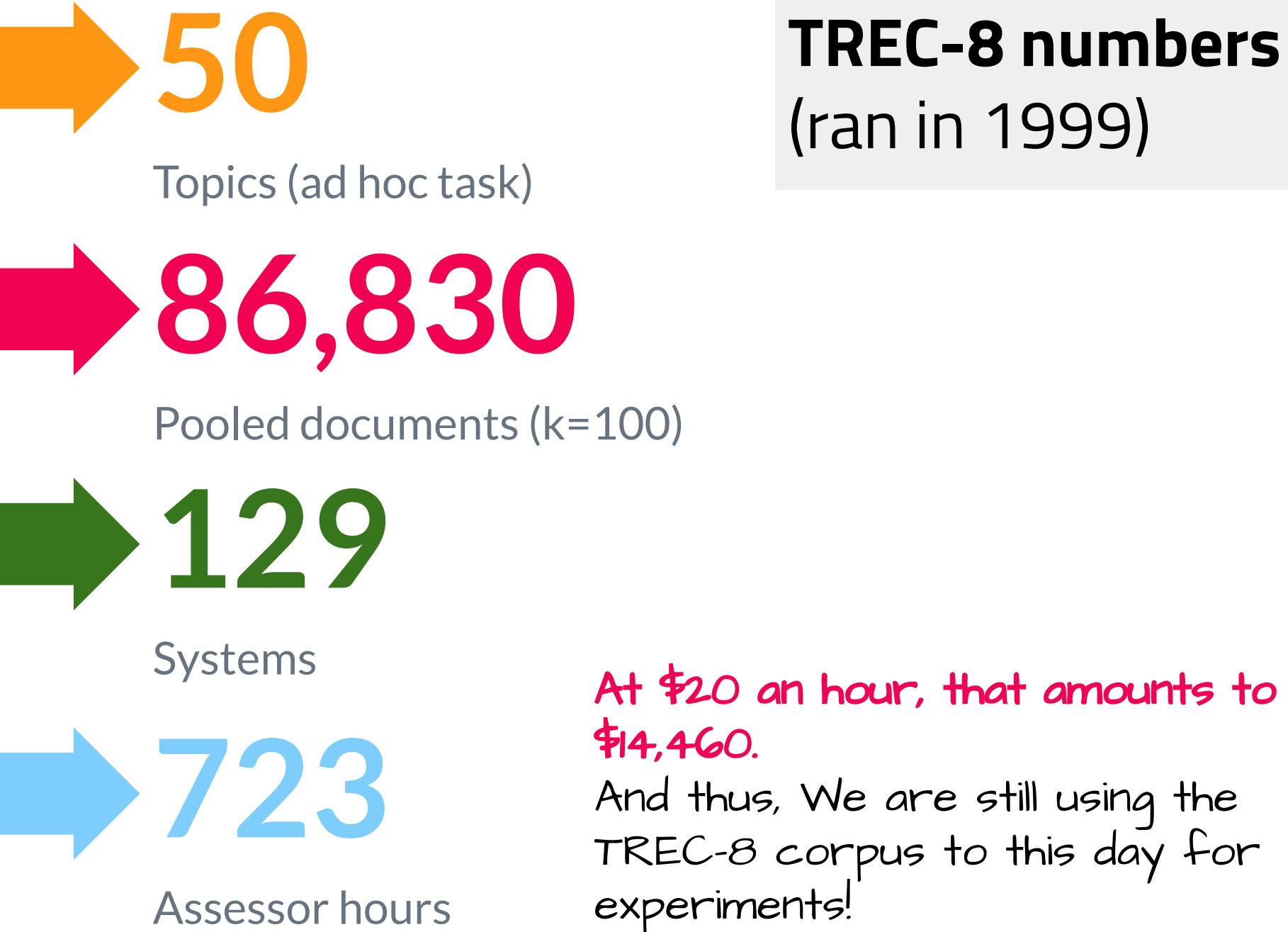
# Explicit vs. implicit

Learning to rank, BM25, LM ...

- Need **training data** to effectively learn the models' (hyper)parameters
- Often explicit relevance judgments are used

Explicit qrels are extremely **expensive** to accumulate  
(can become outdated quickly for dynamic collections)



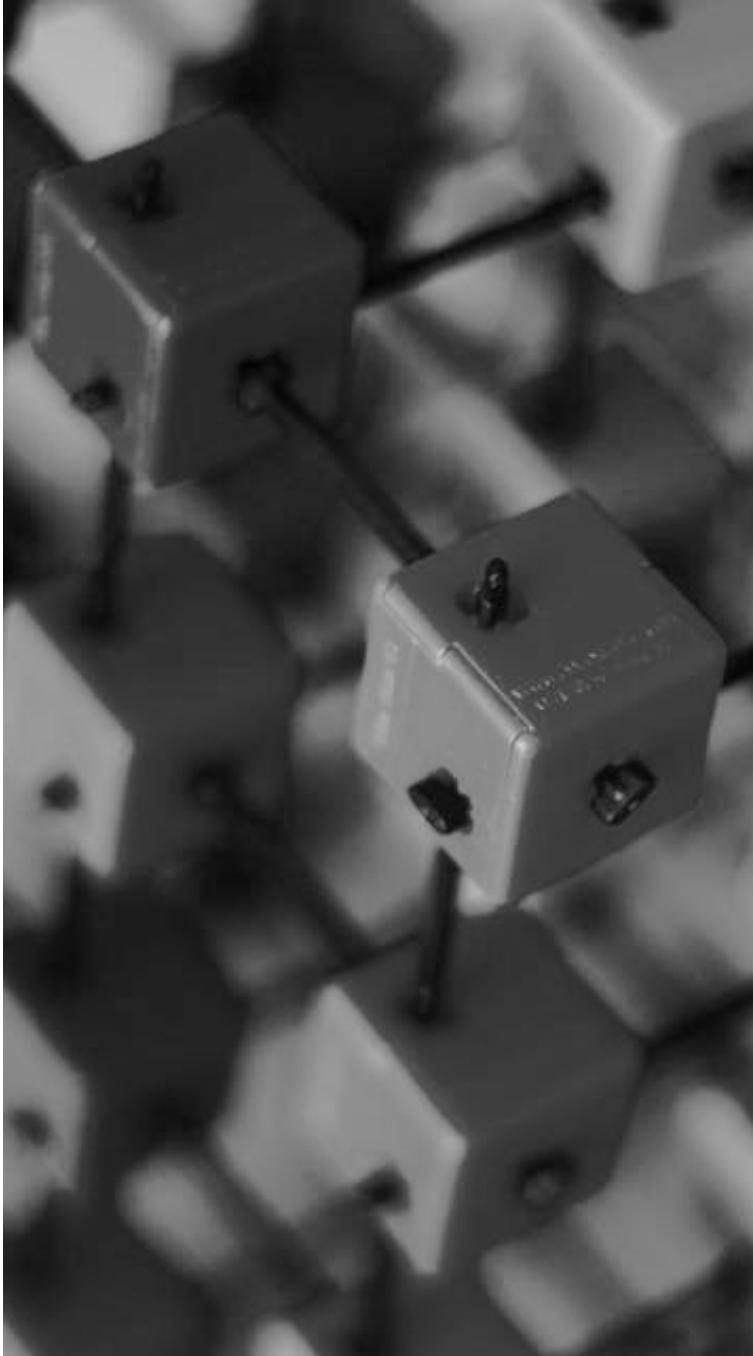


# LETOR features

LEarning TO Rank for Information Retrieval

TF(Term frequency) of body
TF of anchor
TF of title
TF of URL
TF of whole document
IDF(Inverse document frequency) of body
IDF of anchor
IDF of title
IDF of URL
IDF of whole document
TF*IDF of body
TF*IDF of anchor
TF*IDF of title
TF*IDF of URL
TF*IDF of whole document
DL(Document length) of body
DL of anchor
DL of title
DL of URL
DL of whole document
BM25 of body
BM25 of anchor
BM25 of title
BM25 of URL

BM25 of whole document
LMIR.ABS of body
LMIR.ABS of anchor
LMIR.ABS of title
LMIR.ABS of URL
LMIR.ABS of whole document
LMIR.DIR of body
LMIR.DIR of anchor
LMIR.DIR of title
LMIR.DIR of URL
LMIR.DIR of whole document
LMIR.JM of body
LMIR.JM of anchor
LMIR.JM of title
LMIR.JM of URL
LMIR.JM of whole document
PageRank
Inlink number
Outlink number
Number of slash in URL
Length of URL
Number of child page



# Approaches

**Input space** (feature vectors)  
**Output space** (learning target)  
**Hypothesis space**  
**Loss function** (prediction vs. ground truth)

## Pointwise approach

Each document for itself

Input space: feature vector of each doc

Output space: relevance degree of each document

Hypothesis space: functions that take a doc. feature vector as input and output a relevance degree

Regression or classification loss

## Pairwise approach

Each doc. pair for itself

Input space: feature vectors of a pair of docs

Output space: pairwise preferences

Hypothesis space: functions that take a document pair as input and output their relative order

Loss function considers the relative order between the two docs

## Listwise approach

Designed for ranking

Input space:

$$\mathbf{x} = \{x_j\}_{j=1}^m$$

Output space:

(1) relevance degrees of all documents, (2) ranked list of docs

Hypothesis space: functions that take  $\mathbf{x}$  as input and yield (1) or (2)

Loss function considers (1) or (2)

# Approaches

**Input space** (feature vectors)  
**Output space** (learning target)  
**Hypothesis space**  
**Loss function** (prediction vs. ground truth)

## Pointwise approach

Each document for itself.

Input space: feature vector of each doc.

Output space: relevance degree of each document

Hypothesis space: functions that take a doc. feature input and output a relevance degree

Regression or classification loss.

Different loss functions but one and the same evaluation metric (e.g. MAP)

## Pairwise approach

Each doc. pair for itself.

Input space: feature vectors of a pair of docs

Output space: pairwise preferences

Hypothesis space: functions that take a doc. pair as input

and outputs their order

Loss function considers the relative order between the two docs.

## Listwise approach

Designed for ranking.

Input space:

$$\mathbf{x} = \{x_j\}_{j=1}^m$$

Output space: (1) relevance degrees of all documents, (2) ranked list of documents

Hypothesis space: functions that take  $\mathbf{x}$  as input and produce (1) or (2)

Loss function considers (1) or (2)

How to rank a whole set of documents? Another step is needed.

Document ranking is easy.

# L2R categorization

	SVM	Boosting	Neural net	Others
<b>Pointwise</b>		McRank		PRank
<b>Pairwise</b>	RankSVM	RankBoost, LambdaMART, GBRank	RankNet, LambdaRank, FRank	
<b>Listwise</b>	SVM MAP	AdaRank	ListNet	SoftRank, SmoothRank

Tax et al. (2015): "*ListNet, SmoothRank, FenchelRank, FSMRank, LRUF and LARF are Pareto optimal learning to rank methods*"

# Pointwise approach

Direct application of  
standard supervised ML.

# Pointwise categories

- Regression based algorithms
  - Real-valued relevance scores
- Classification based algorithms
  - Non-ordered categories
- Ordinal regression based algorithms
  - Variables with a natural categorical ordering





- Main issue of regression?

# Polynomial regression function

Given  $q$  and associated  $\mathbf{x} = \{x_j\}_{j=1}^m$ ,

let the ground truth label be:

**binary:**  $\vec{y}_j = (0, 1)$  or  $\vec{y}_j = (1, 0)$

ordered categories:  $\vec{y_j} = (0, 0, \dots, 1, \dots, 0)$

doc judged belonging to a category (e.g. 'Fair' or 'Good')

*Scoring function:*  $\vec{f} = (f_1, f_2, \dots, f_k)$  with Predictor of  $k$ th element in the ground truth vec.  
 $f_k$  T<sup>th</sup> feature in feature vector  $i$

$$\begin{aligned} \underline{f_k(x_j)} &= w_{k,0} + w_{k,1} \times x_{j,1} + \dots + w_{k,T} \times x_{j,T} \\ &+ w_{k,T+1} \times x_{j,1}^2 + w_{k,T+2} \times x_{j,1} \times x_{j,2} + \dots \end{aligned}$$

Combination coefficient

## Combination coefficient

# Pairwise approach

# Pairwise

Focus: **relative ordering** of pairs of documents

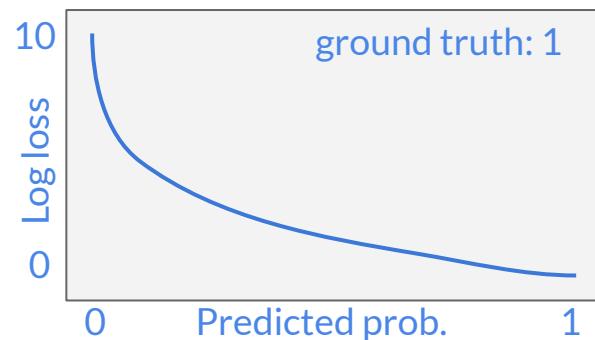
Ranking problem **reduced** to a classification problem  
(goal: minimize #misclassified pairs)

IR issue: **target cost** vs. **optimization cost** (tractable opt.,  
good approximation of target cost)

Training data:

$$\{(x_1, x_2, +1), (x_1, x_3, -1), \dots, (x_i, x_j, +1), \dots\}$$

# RankNet



Given  $q$  and two documents  $x_u$  and  $x_v$ ,

**modeled prob:** 
$$P_{u,v}(f) = \frac{\exp(f(x_u) - f(x_v))}{1 + \exp(f(x_u) - f(x_v))}$$

Based on the diff. between  
the two documents' scores

Shallow neural network  
learns scoring function  $f$ ;  
gradient descent as  
optimization alg

$$L(f; x_u, x_v, y_{u,v}) = -\bar{P}_{u,v} \log P_{u,v}(f)$$

Cross-entropy loss

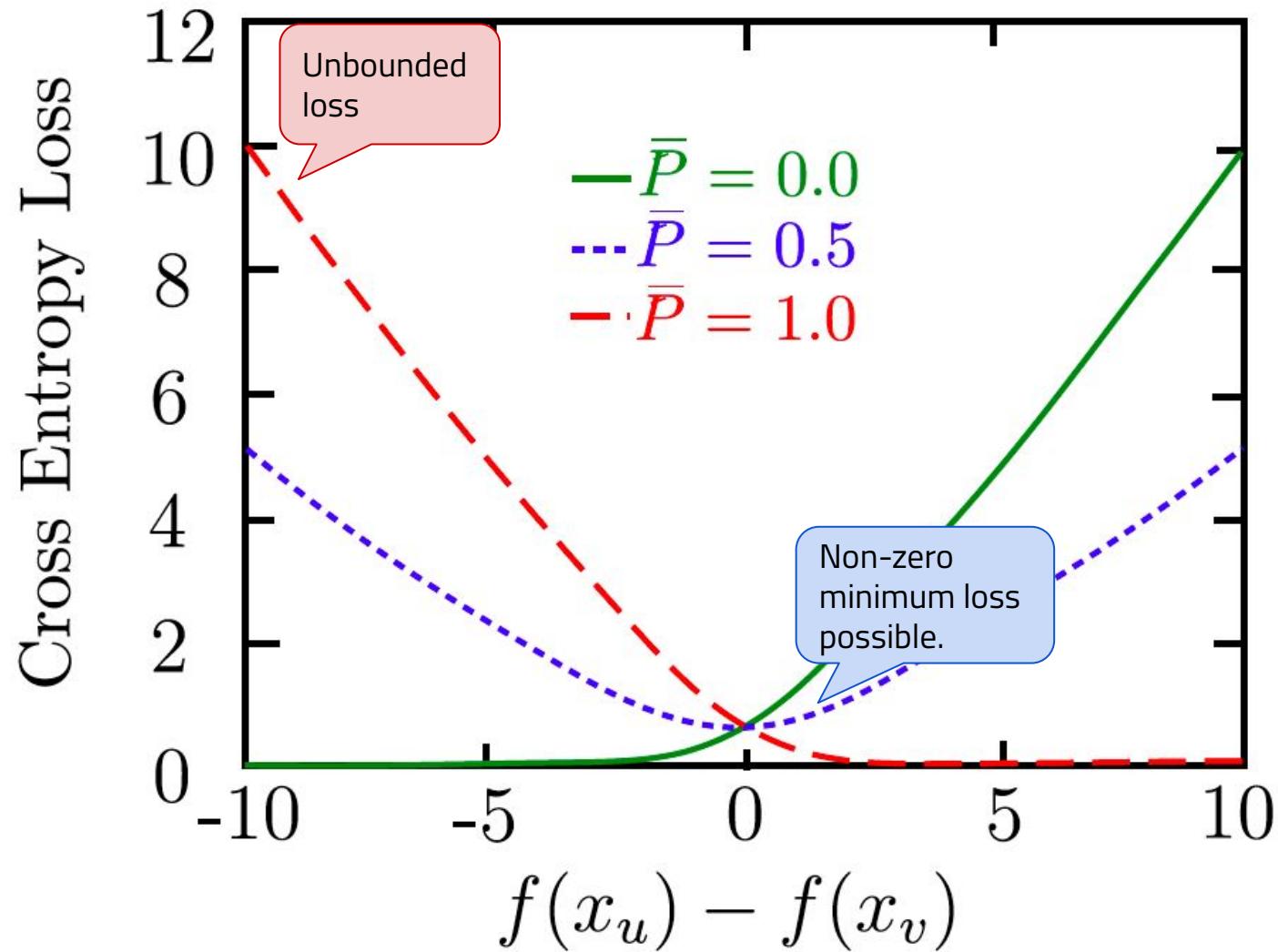
$$-\left(1 - \bar{P}_{u,v}\right) \log(1 - P_{u,v}(f))$$

Target probability:

$$\bar{P}_{u,v} = 1, \text{ if } y_{u,v} = 1;$$

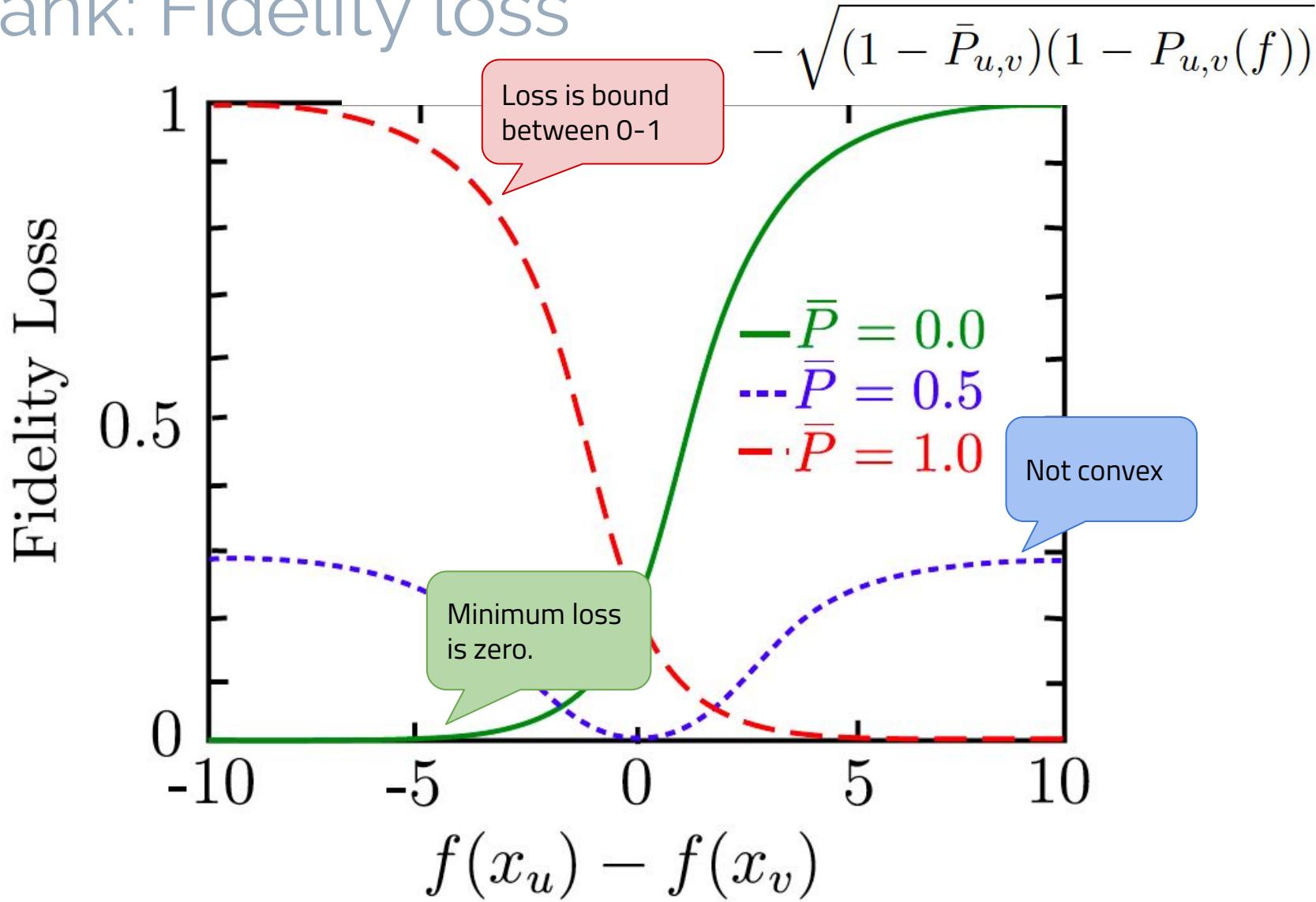
$$\bar{P}_{u,v} = 0 \text{ otherwise}$$

# RankNet - loss function issue

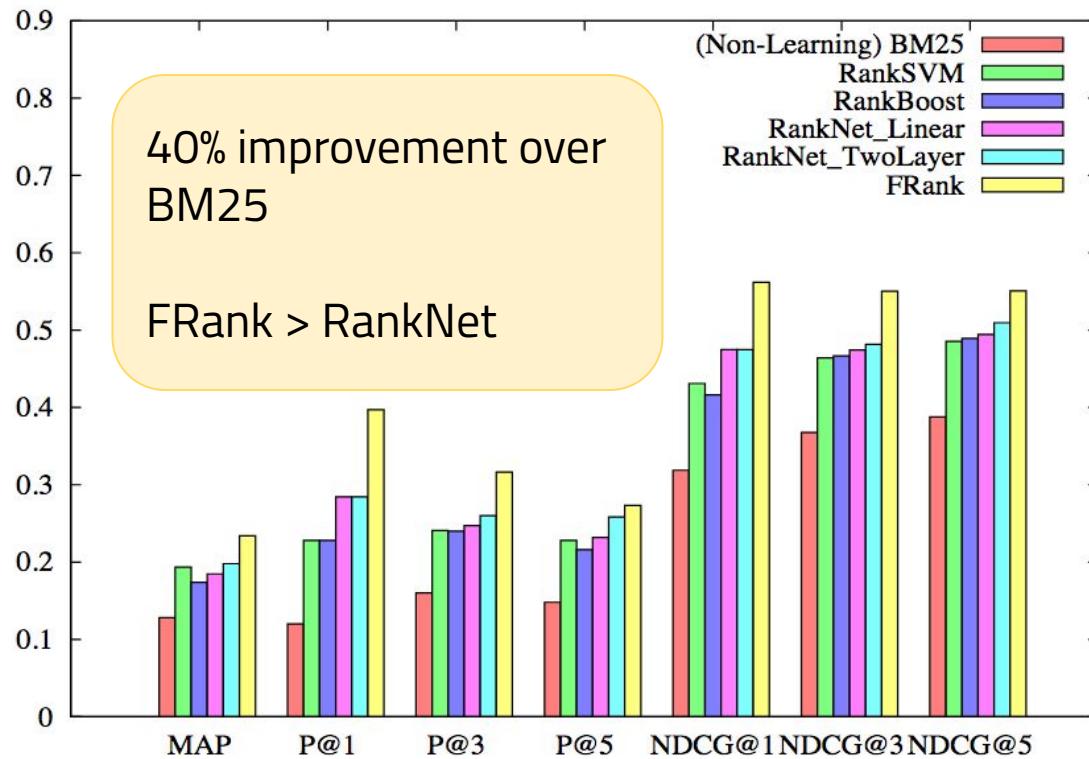


$$L(f; x_u, x_v, y_{u,v}) = 1 - \sqrt{\bar{P}_{u,v} P_{u,v}(f)}$$

## FRank: Fidelity loss



# Experimentally: RankNet vs. FRank



**TREC topic distillation** aims at finding key resources which are high-quality pages for certain topics.

Corpus: 1M .gov pages, 14 features per document

50 topics (between 1 and 86 relevant docs per topic)

4-fold cross validation



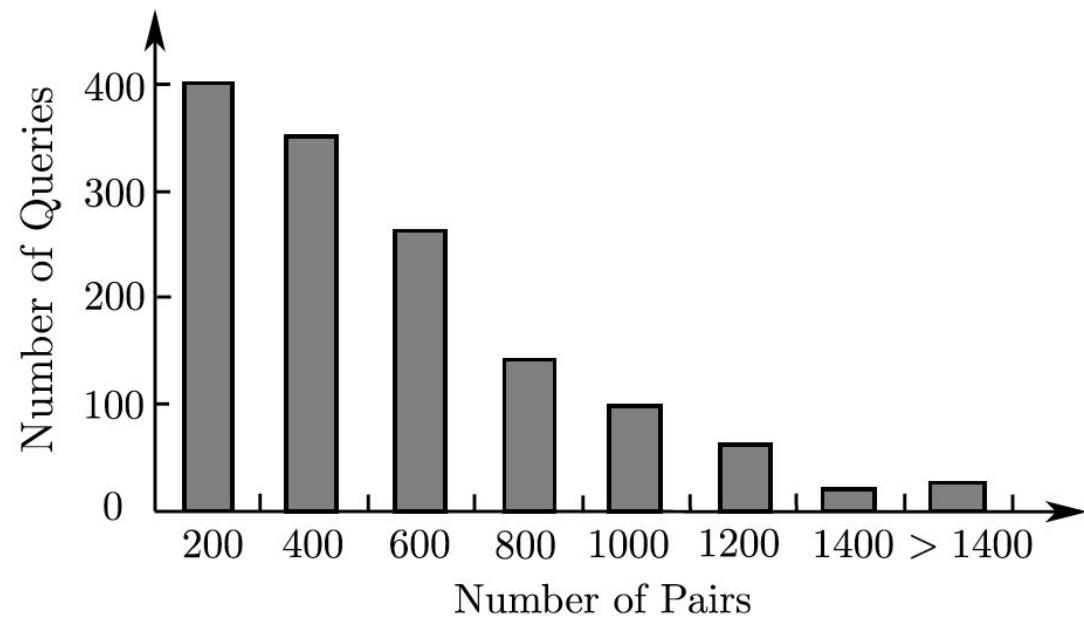
Is training of the pairwise approach slower/faster than the pointwise one?

# Document pair issue

Document pairs only make it into the training set if their relevance degrees differ

Queries differ widely in the number of pairs they generate

Loss will be dominated by queries with many pairs



# Beyond RankNet ... [6]

## >> LambdaRank

- Insight: neural net training requires only the *gradients* of the cost function
- Heuristic rules on how the cost changes if document rankings are swapped

## >> LambdaMART

- Gradient boosted decision trees



# Listwise approach

# Listwise

- 1) Optimize a continuous & differentiable approximation of an IR metric.
- 2) Optimize a continuous & differentiable bound of an IR metric.
- 3) Choose optimization approach that can handle complex objectives.

## Direct optimization



Output space contains relevance degrees of all docs associated with q.

Loss function **optimizes an IR metric.**

Not easy as MAP, NDCG, ... are non-continuous & non-differentiable.

Most optimization techniques are designed for continuous and differentiable functions.

## Permutation-based

The output space contains the permutation of the documents associated with q.

The loss function measures the difference between the permutation given by the hypothesis and the ground truth permutation.

Example: ListNet

# ListNet

Required: a loss function that considers the document list

Idea: define two probability distributions, one on the hypothesized and one on the reference ranking. Use a metric that **compares the two probability distributions** as loss function.



Which metric?

Given scoring function  $f$

and document relevance scores  $\mathbf{s} = \{s_j\}_{j=1}^m$ , where  $s_j = f(x_j)$ ,

define a prob. for each possible permutation  $\pi$  of the documents:

$$P(\pi|\mathbf{s}) = \prod_{j=1}^m \frac{\varphi(s_{\pi^{-1}(j)})}{\sum_{u=j}^m \varphi(s_{\pi^{-1}(u)})}$$

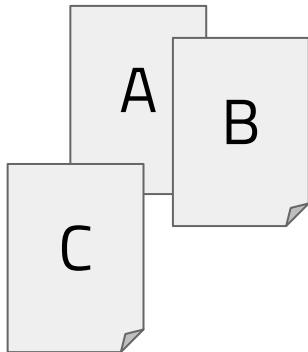
permutation probability

doc. ranked at jth position in the permutation

transformation function (e.g. exponential)

conditional probability

# ListNet permutation example



Given 3 documents for query q, what is the probability of the ranking permutation A-B-C?

$$P_\pi = P_1 \times P_2 \times P_3$$

$$P_1 = \frac{\varphi(s_A)}{\varphi(s_A) + \varphi(s_B) + \varphi(s_C)}$$

Probability of doc A being ranked at the top.  
Determined by comparing A's score to B's and C's scores.

$$P_2 = \frac{\varphi(s_B)}{\varphi(s_B) + \varphi(s_C)}$$

Probability of B being ranked at position 2, given that A has been ranked already.

$$P_3 = 1 \quad \text{Only C is left.}$$

# ListNet

ListNet defines the permutation probability distribution based on the scores given by the scoring function.

The **reference permutation probability distribution** is based on the ground truth labels.

**KL divergence** between both distributions to define the listwise ranking loss:

$$L(f; \mathbf{x}, \pi_y) = D(P(\pi | \varphi(f(w, \mathbf{x}))) \| P_y(\pi))$$

Shallow neural network learns scoring function  $f$ ; gradient descent as optimization alg

Practical issue: training over all possible permutations of a list is impractical ( $m!$  permutations of size  $m$ )

# Benchmarks & practice

	Queries	Doc.	Rel.	Feat.	Year
LETOR 3.0 – Gov	575	568 k	2	64	2008
LETOR 3.0 – Ohsumed	106	16 k	3	45	2008
LETOR 4.0	2,476	85 k	3	46	2009
Yandex	20,267	213 k	5	245	2009
Yahoo!	36,251	883 k	5	700	2010
Microsoft	31,531	3,771 k	5	136	2010

Burges et al. (2011) used a linear combination of 12 ranking models, 8 of which were LambdaMART (Burges, 2010) boosted tree models, 2 of which were LambdaRank neural nets, and 2 of which were logistic regression models. While LambdaRank was originally instantiated using neural nets, LambdaMART implements the same ideas using the boosted-tree style MART algorithm, which itself may be viewed as a gradient descent algorithm. Four of the LambdaMART rankers (and one of the nets) were trained using the ERR measure, and four (and the other net) were trained using NDCG. Extended training sets were also generated by randomly deleting feature vectors for each query.

[8] Chapelle, Olivier, and Yi Chang. "Yahoo! learning to rank challenge overview." Proceedings of the learning to rank challenge. 2011.

# Implicit feedback example



# Potential source: clickthrough data

RQ: How effective is **implicit feedback** in a large-scale web environment?

- Web search engines use thousands of features and are heavily tuned
- Tuning is a continuous process
- Long-tail issues (no feedback for rare queries)

RQ: How can implicit feedback be **combined** with the existing ranking produced by the search system?

- Reranking vs. RankNet

RQ: Can we move beyond clicks?

# Clickthrough data in L2R

- 1) Derive a set of features from implicit feedback
- 2) At runtime, the search engine needs to fetch the implicit feedback features associated with each (query, URL) pair

L2R needs to be robust to missing values: *long tail* issue

Here: **RankNet**

- Neural net based tuning algorithm that optimizes feature weights to best match explicitly provided **pairwise** user preferences
- Has both train- and run-time efficiency
- Aggregate (query, URL) pair features across all instances in the session logs

# Features

Different types of user action features

Directly observed vs.  
derived features  
(derivations)

Browsing behaviour after  
the result has been clicked

Snippet based features are  
included as users often  
determine relevance based  
on snippet information

## *Clickthrough features*

Position	Position of the URL in Current ranking
ClickFrequency	Number of clicks for this query, URL pair
ClickProbability	Probability of a click for this query and URL
ClickDeviation	Deviation from expected click probability
IsNextClicked	1 if clicked on next position, 0 otherwise
IsPreviousClicked	1 if clicked on previous position, 0 otherwise
IsClickAbove	1 if there is a click above, 0 otherwise
IsClickBelow	1 if there is click below, 0 otherwise

## *Browsing features*

TimeOnPage	Page dwell time
CumulativeTimeOnPage	Cumulative time for all subsequent pages after search
TimeOnDomain	Cumulative dwell time for this domain
TimeOnShortUrl	Cumulative time on URL prefix, no parameters
IsFollowedLink	1 if followed link to result, 0 otherwise
IsExactUrlMatch	0 if aggressive normalization used, 1 otherwise
IsRedirected	1 if initial URL same as final URL, 0 otherwise
IsPathFromSearch	1 if only followed links after query, 0 otherwise
ClicksFromSearch	Number of hops to reach page from query
AverageDwellTime	Average time on page for this query
DwellTimeDeviation	Deviation from average dwell time on page
CumulativeDeviation	Deviation from average cumulative dwell time
DomainDeviation	Deviation from average dwell time on domain

## *Query-text features*

TitleOverlap	Words shared between query and title
SummaryOverlap	Words shared between query and snippet
QueryURLOverlap	Words shared between query and URL
QueryDomainOverlap	Words shared between query and URL domain
QueryLength	Number of tokens in query
QueryNextOverlap	Fraction of words shared with next query

# Evaluation

Random sample of queries from a Microsoft query log with associated results and traces of user actions

8 weeks of user interactions with 1.2M unique queries (sufficient interactions for 50% of queries) and 12M interactions

On average, 30 results judged per query by human assessors on a **six point scale** (83K results judged)

	<b>MAP</b>
BM25F (content + link-based info)	0.184
RankNet	0.215
<b>ReRanking</b>	
BM25F + Click-through statistics only	0.215
BM25F + Implicit feedback	0.222
<b>Integrated as features</b>	
RankNet + Implicit feedback	<u>0.248</u>

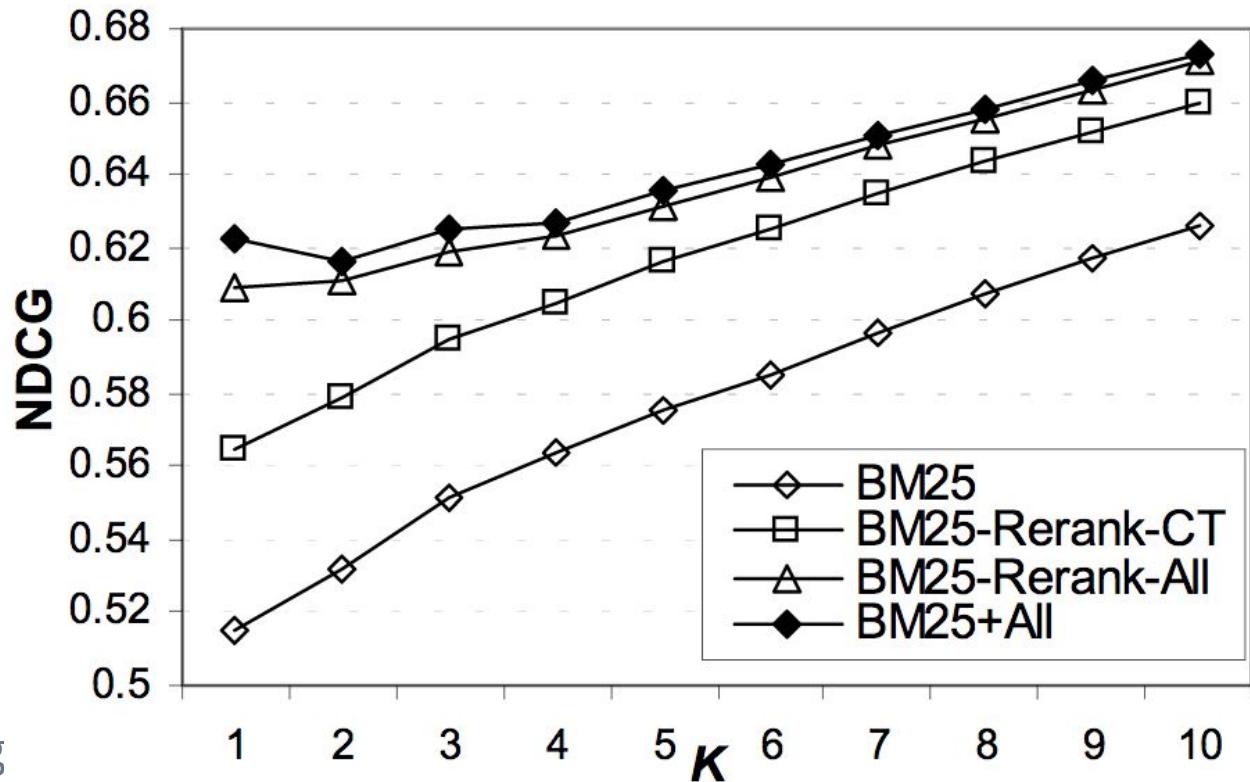
# Results

BM25F: content-based (fields) and query-independent link-based information (PageRank, URL depth, etc.); does *not* make use of implicit/explicit feedback

BM25F-RerankCT:  
reranking based on  
clickthrough statistics  
(weight  $w=1000$ )

BM25F-RerankAll:  
RankNet-based  
reranking with all  
behavioural features

BM25F+All:  
RankNet-based ranking  
on BM25F features+IF



# Implicit feedback can replace hundreds of features

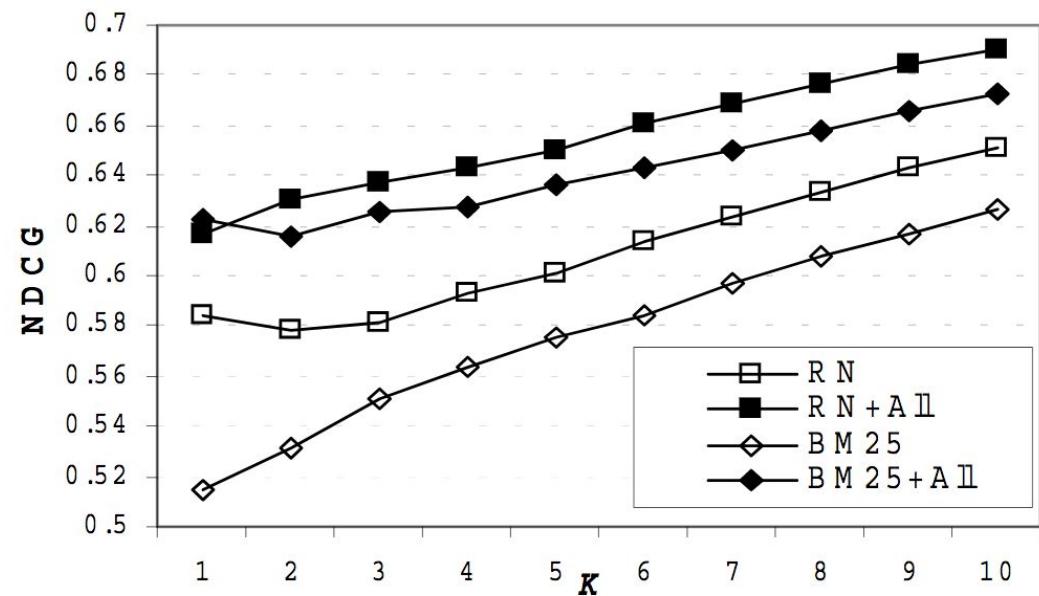
RankNet (RN) : hundreds of features of a Web search engine; based on explicit judgments

RankNet+All: including IF features

BM25F: content-based (fields) and query-independent link-based information (PageRank, URL depth, etc.)

BM25F+All: train RankNet over the feature set of BM25F and IF

	<b>MAP</b>
BM25F	0.184
BM25F-Rerank-CT	0.215
BM25F-RerankImplicit	0.218
BM25F+Implicit	<b>0.222</b>
RN	0.215
RN+All	<b>0.248</b>



# Software (actively maintained)

## **RankLib**

[sourceforge.net/p/lemur/wiki/RankLib](http://sourceforge.net/p/lemur/wiki/RankLib)

## **TF-Ranking**

[github.com/tensorflow/ranking](https://github.com/tensorflow/ranking)

(ICTIR/SIGIR 2019 tutorials on the toolkit)

## **LIBLINEAR**

[www.csie.ntu.edu.tw/~cjlin/liblinear/](http://www.csie.ntu.edu.tw/~cjlin/liblinear/)

## **XGBoost**

<https://github.com/dmlc/xgboost>

# Summary L2R

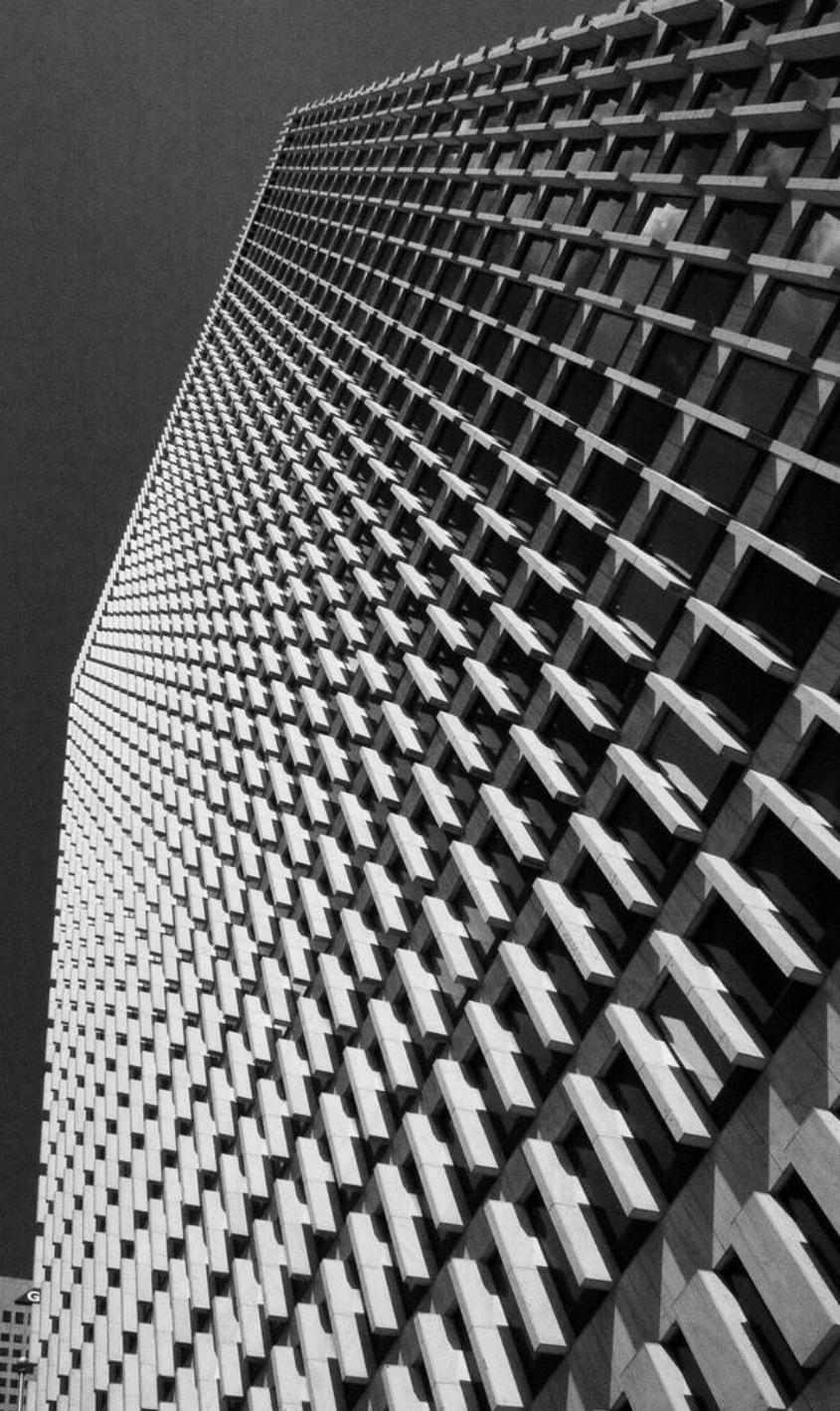
It is not sufficient to deploy standard ML algs; need to be adapted towards ranking (pairwise, listwise)

Listwise vs. pairwise: the former makes most theoretical sense, the latter is empirically robust and efficient

L2R is often neural-network based (shallow, not deep)

Best results achieved in ensembles

Current research: unbiased L2R, efficient & scalable L2R, direct metric optimization



# Neural IR

[10] Mitra, Bhaskar, and Nick Craswell. "An introduction to neural information retrieval." *Foundations and Trends® in Information Retrieval* 13.1 (2018): 1-126.



Ian Goodfellow @goodfellow\_ian · 26 Mar 2018

Reviewers should be very suspicious of anyone who has implemented their own baseline. There are a lot of subtle ways to screw up deep learning algorithms and authors have an incentive not to check their own baseline very carefully.

1

6

18



Ian Goodfellow

@goodfellow\_ian

Following

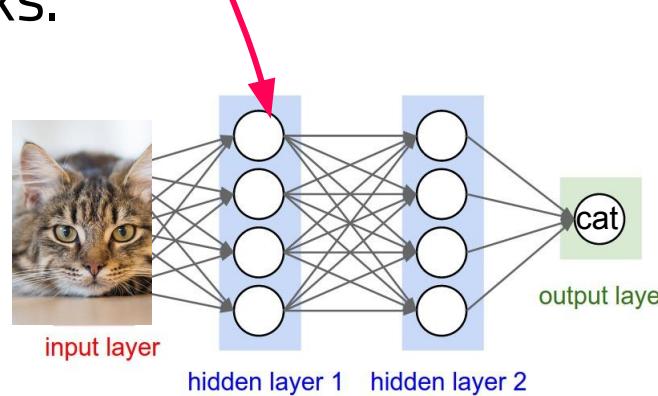
Usually, at least one of the baselines should be a result published in another paper, where the authors of that other paper had some incentive to get a good result. This way the evaluations are at least incentive-compatible.

8:34 PM - 26 Mar 2018

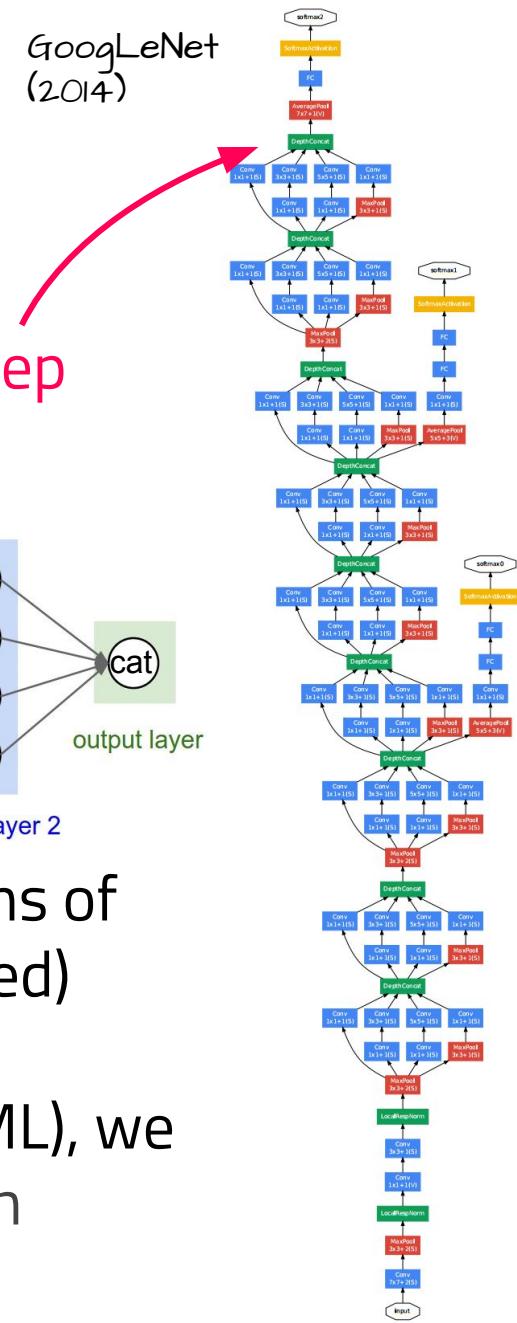
Different fields have different takes on reproducibility ...

# Overview

Neural IR is the application of **shallow** or **deep** neural networks to IR tasks.



GoogLeNet  
(2014)



#params  
LM, BM25

Neural IR models contain thousands/millions of parameters (large-scale training data needed)

Instead of hand-crafting **features** (classic ML), we now handcraft NN **architectures** and search for/generate architectures automatically

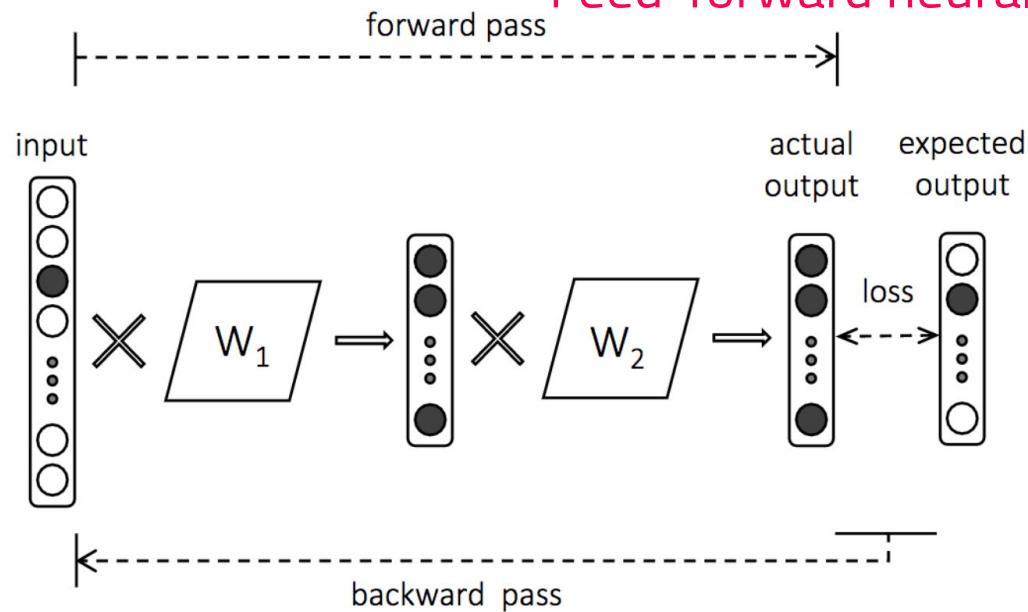
[11] Szegedy, Christian, et al. "Going deeper with convolutions." CVPR. 2015.

[12] <http://cs231n.github.io/neural-networks-1/>

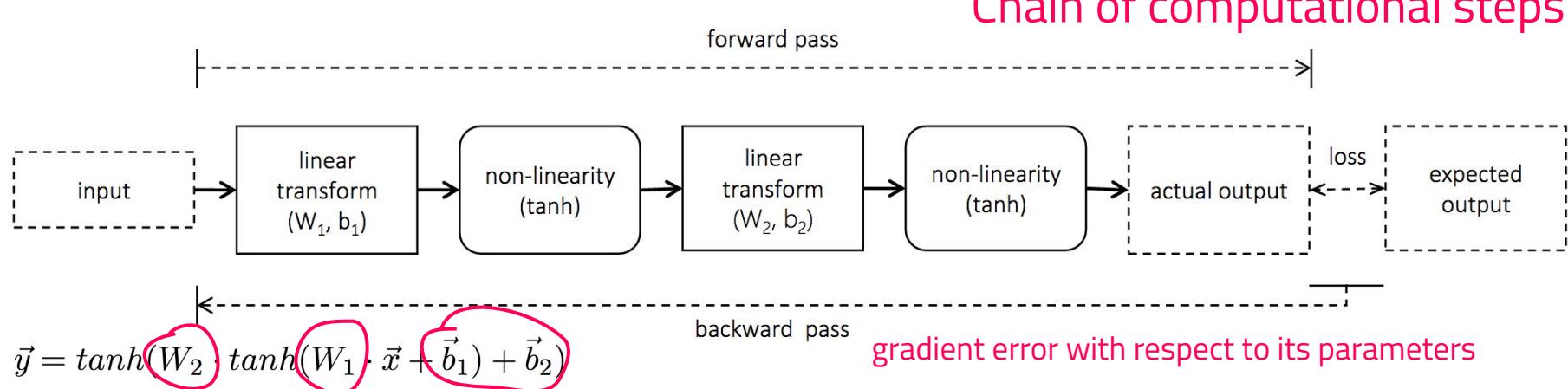
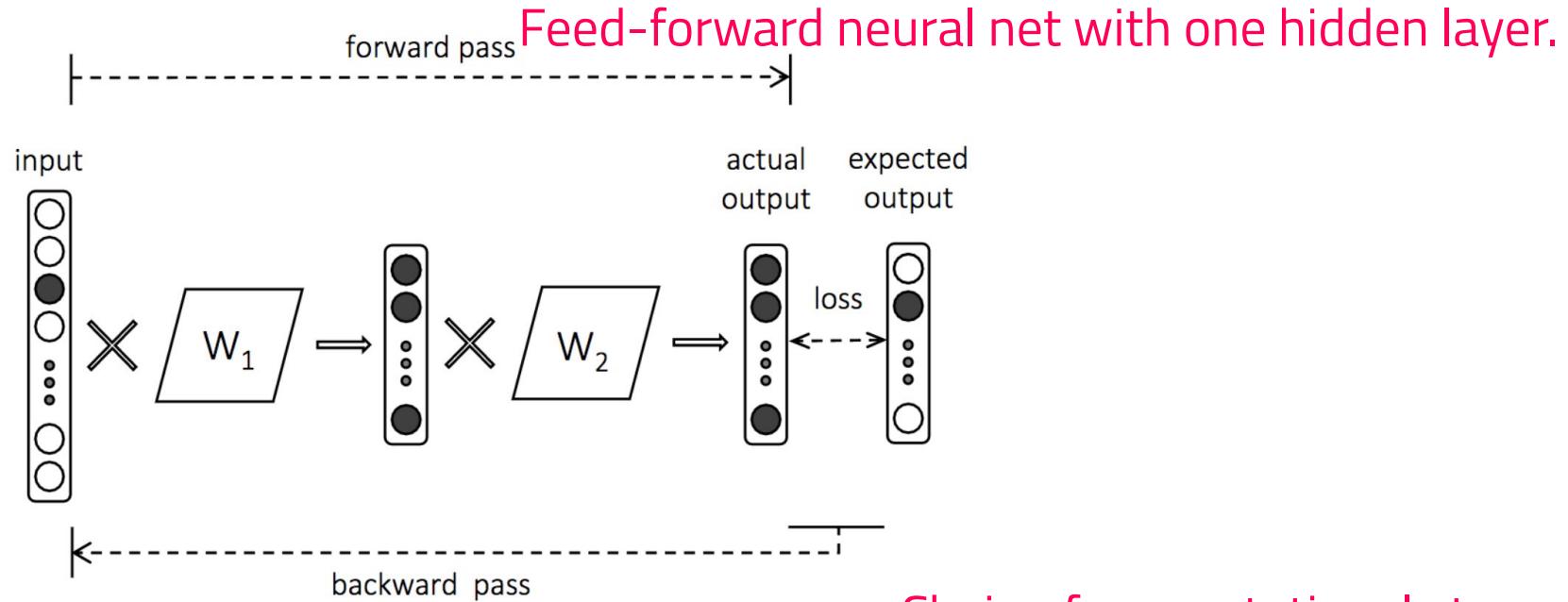
Image sources: [11,12]

# Neural net basics

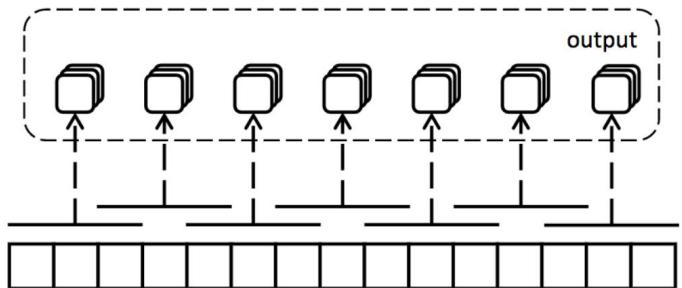
Feed-forward neural net with one hidden layer.



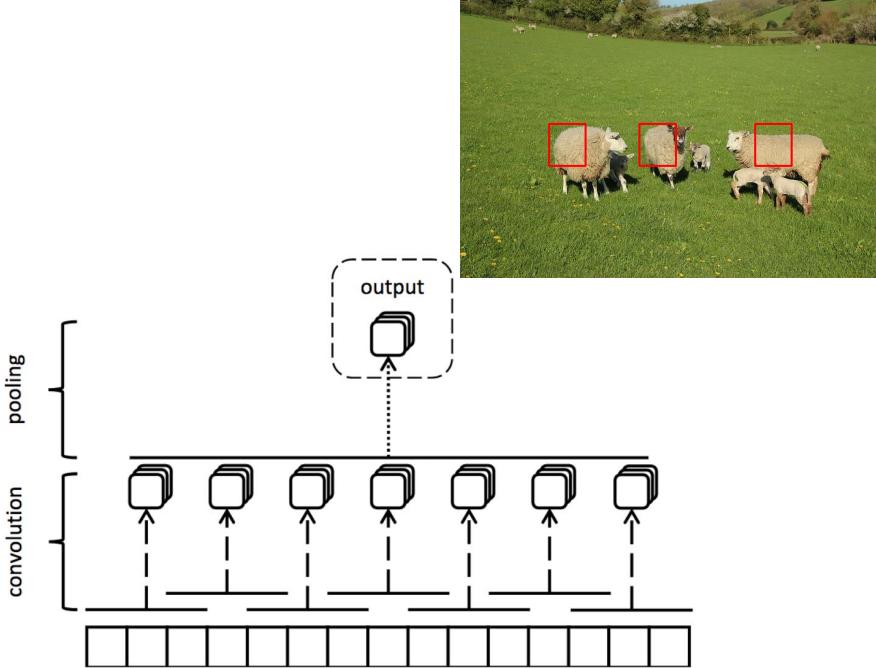
# Neural net basics



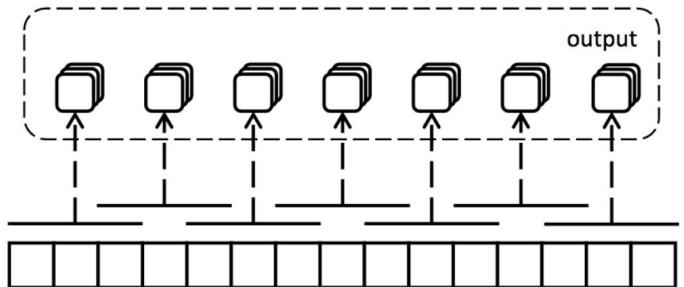
# CNNs and RNNs



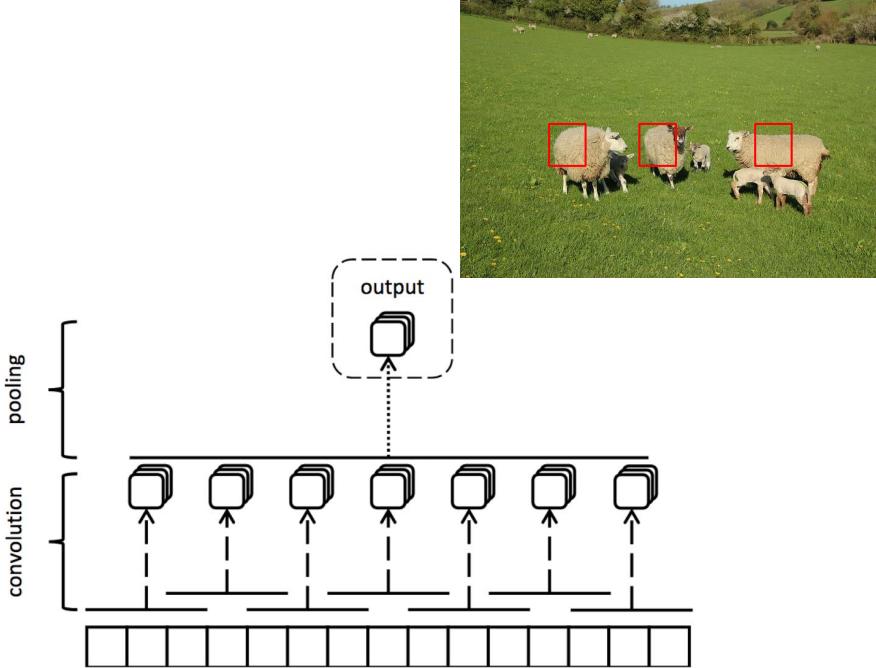
CNN: convolutional neural network.  
Most often found in computer vision setups.



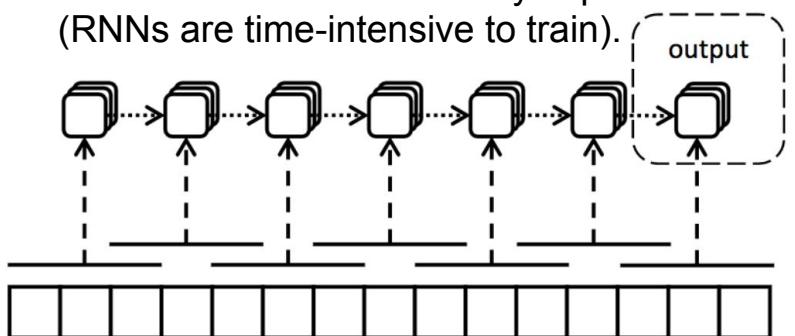
# CNNs and RNNs



CNN: convolutional neural network.  
Standard in computer vision setups.



RNN: recurrent neural network.  
Often found in NLP setups, but  
alternatives are continuously explored  
(RNNs are time-intensive to train).



All shift-invariant neural operations,  
including convolutional and recurrent  
layers move a fixed size window over  
the input space with fixed stride.

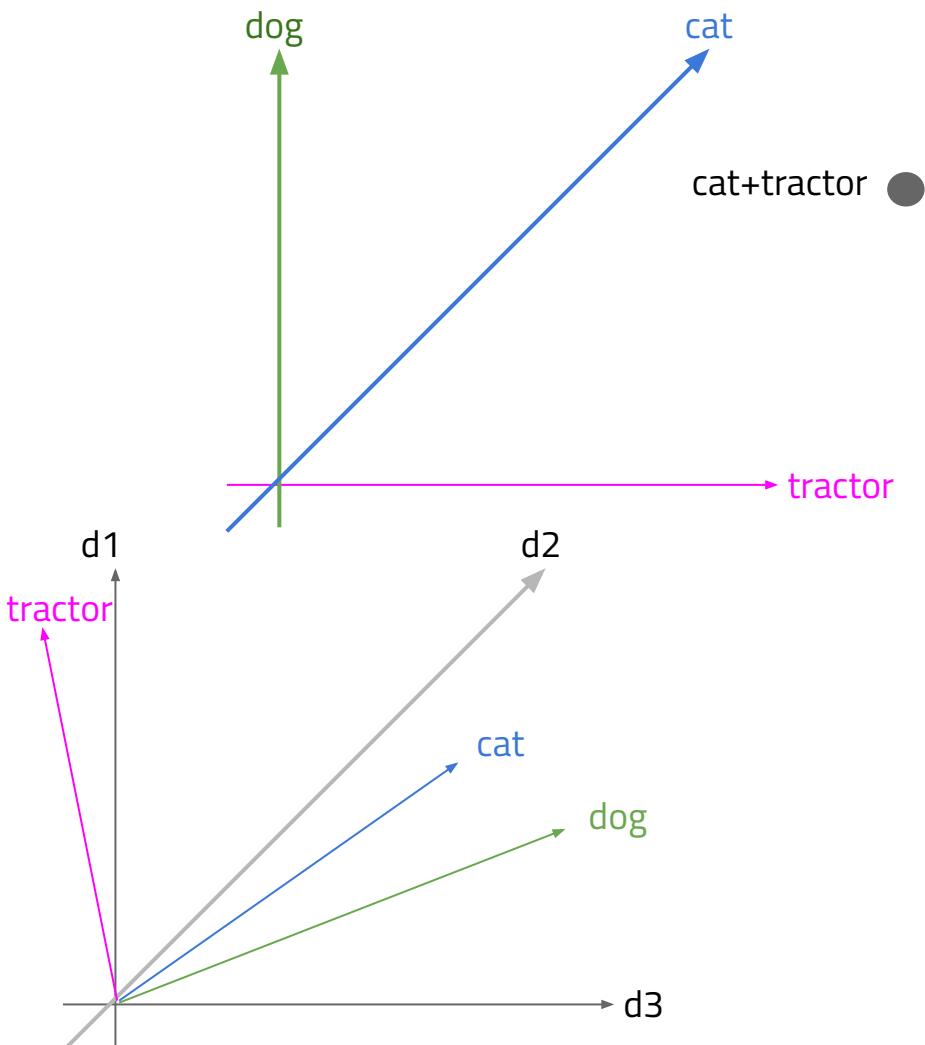
Each window is projected by a  
parameterized neural operation, often  
followed by an aggregation step such  
as (max) pooling.

*Obama entered the White House. He met there with Trump.*

Image source: [10]

# Text representations

# Text representations



In DL, commonly known as  
“one-hot” encoding/repr.

Local vector representation  
(sparse, high-dimensional)

$$cat = (0, 0, 0, 0, 1, 0, 0, 0, 0, 0, \dots, 0)$$

$$tractor = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, \dots, 0)$$

$$dog = (1, 0, 0, 0, 0, 0, 0, 0, 0, 0, \dots, 0)$$

$$feline = (0, 0, 0, 0, 0, 0, 0, 0, 0, 1, \dots, 0)$$

**Semantic gap:** the similarity between any of those two terms is zero.

Handcrafted or learnt. Individual dimensions no longer interpretable.

distributed vector representation  
(dense, often low-dimensional)

$$cat = (0, 1, 1, 0, 1, 0, 0, 0, 0, 0)$$

$$tractor = (0, 0, 0, 1, 0, 0, 0, 1, 1, 0)$$

$$dog = (1, 1, 0, 0, 0, 0, 0, 0, 0, 0)$$

$$feline = (0, 1, 1, 1, 1, 0, 0, 0, 0, 1)$$

The similarity between some of those terms is no longer zero.

# Text representations

Text representations can be learnt in a **supervised** or **unsupervised** fashion.

In IR, supervised approaches use query-document pairs.

In IR, the unsupervised approach uses just queries or just documents.

**"Similarity"** is not an absolute concept, it depends on the task & context at hand. In IR, it is related to relevance.

In DL, commonly known as "one-hot" encoding/repr.

**Local vector representation**  
(sparse, high-dimensional)

$cat = (0, 0, 0, 0, 1, 0, 0, 0, 0, 0, \dots, 0)$

$tractor = ($  Individual dimensions are atomic and interpretable.  $0)$

$dog = (1, 0, 0, 0, 0, 0, 0, 0, 0, 0, \dots, 0)$

$feline = (0, 0, 0, 0, 0, 0, 0, 0, 0, 1, \dots, 0)$

**Semantic gap:** the similarity between any of those two terms is zero.

Handcrafted or learnt. Individual dimensions no longer interpretable.

**distributed vector representation**  
(dense, often low-dimensional)

$cat = (0, 1, 1, 0, 1, 0, 0, 0, 0, 0)$

$tractor = (0, 0, 0, 1, 0, 0, 0, 1, 1, 0)$

$dog = (1, 1, 0, 0, 0, 0, 0, 0, 0, 0)$

$feline = (0, 1, 1, 1, 1, 0, 0, 0, 0, 1)$

The similarity between some of those terms is no longer zero.

# Feature-based representation examples

*in-document features*

$$cat = (0, 0, \underset{doc 3}{\downarrow} 1, 0, \underset{doc 5}{\downarrow} 1, 0, 0, 0, 1, 0, \dots, \underset{doc 1678}{\downarrow} 1, 0)$$

*neighbouring-word features*

$$cat = (0, 0, \underset{runs}{\downarrow} 1, 0, \underset{eats}{\downarrow} 1, 0, 0, 0, \underset{hurt}{\downarrow} 1, 0, \dots, \underset{is}{\downarrow} 1, 0)$$

*neighbouring-word with distances features*

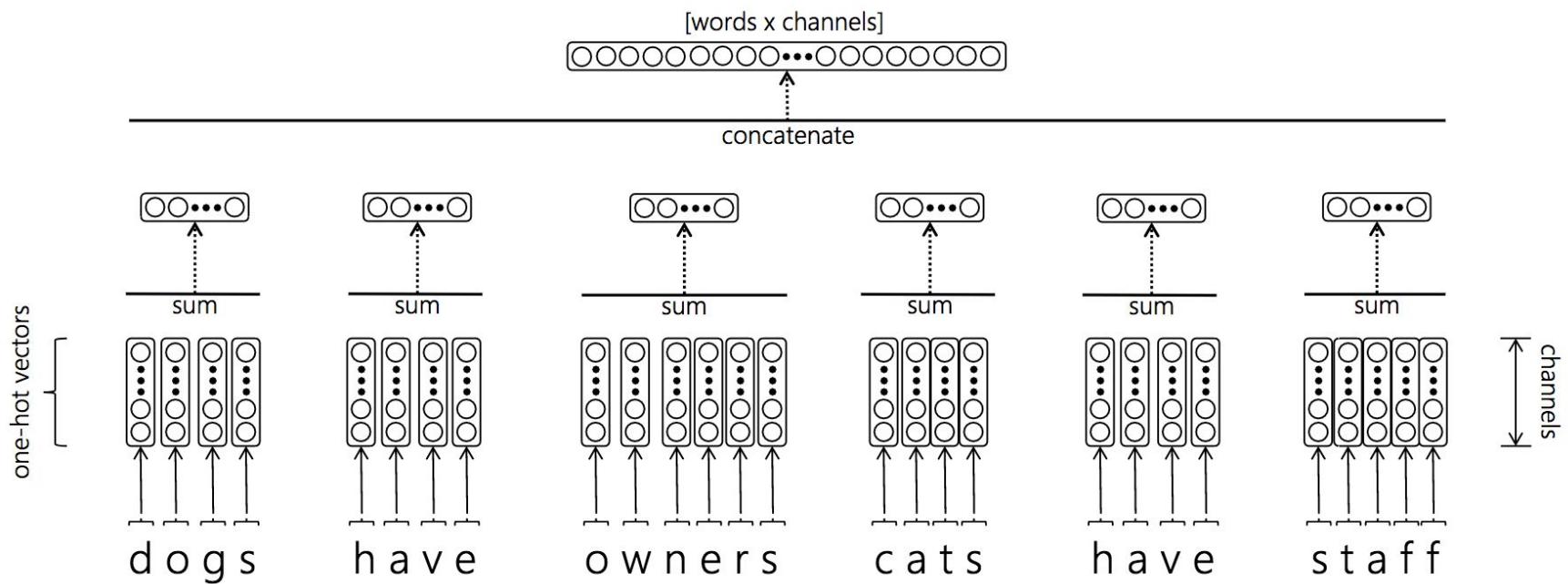
$$cat = (0, 0, \underset{runs^{+1}}{\downarrow} 1, 0, \underset{black^{-1}}{\downarrow} 1, 0, 0, 0, \underset{along^{+2}}{\downarrow} 1, 0, \dots, \underset{old^{-1}}{\downarrow} 1, 0)$$

*character trigram features*

$$kitten = (0, 0, \underset{en\#}{\downarrow} 1, 0, \underset{\#ki}{\downarrow} 1, 0, \underset{tte}{\downarrow} 1, 0, \underset{kit}{\downarrow} 1, 0, \underset{ten}{\downarrow} 1, 0, \dots, \underset{itt}{\downarrow} 1, 0)$$

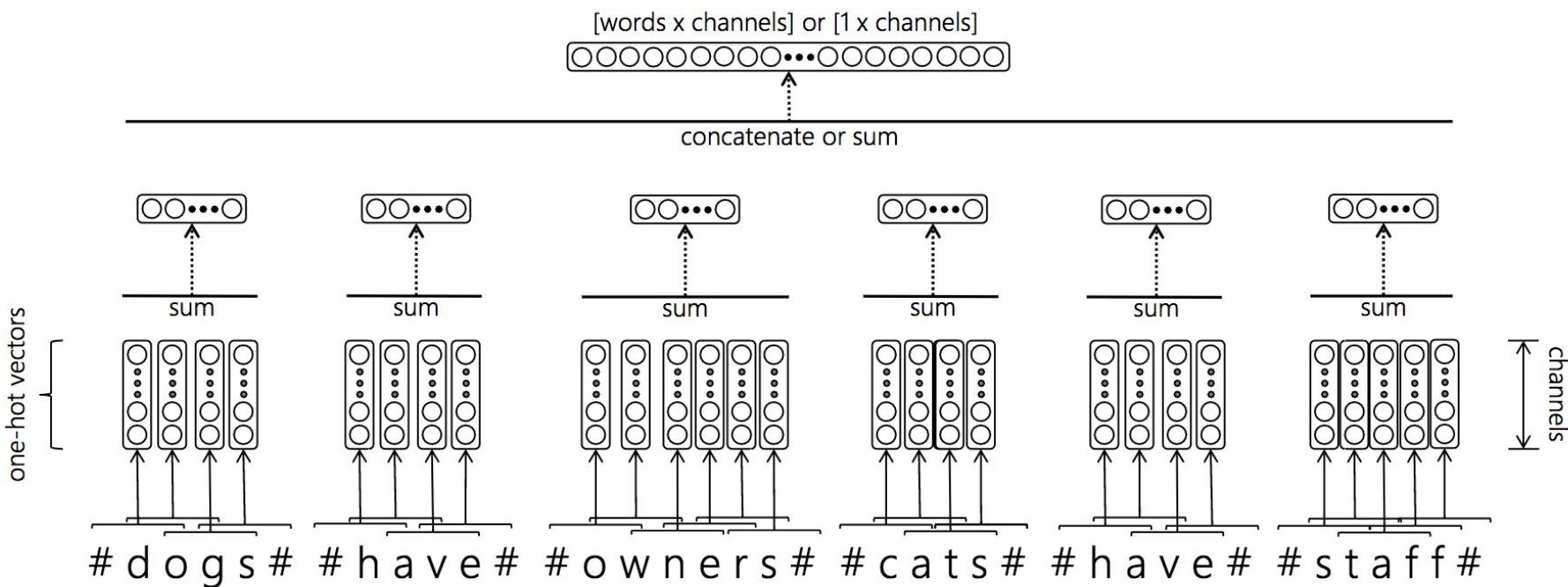
# Text input to deep neural nets

term-level input with  
bag-of-chars per term



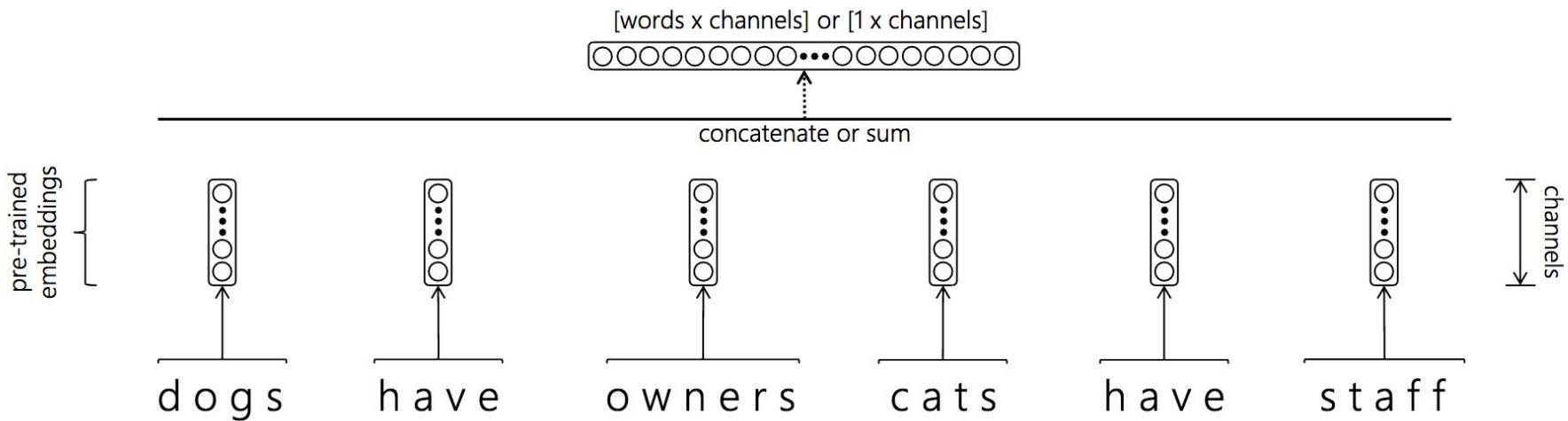
# Text input to deep neural nets

term-level input with  
bag-of-trigrams per term



# Text input to deep neural nets

term-level input with  
pre-trained word embeddings

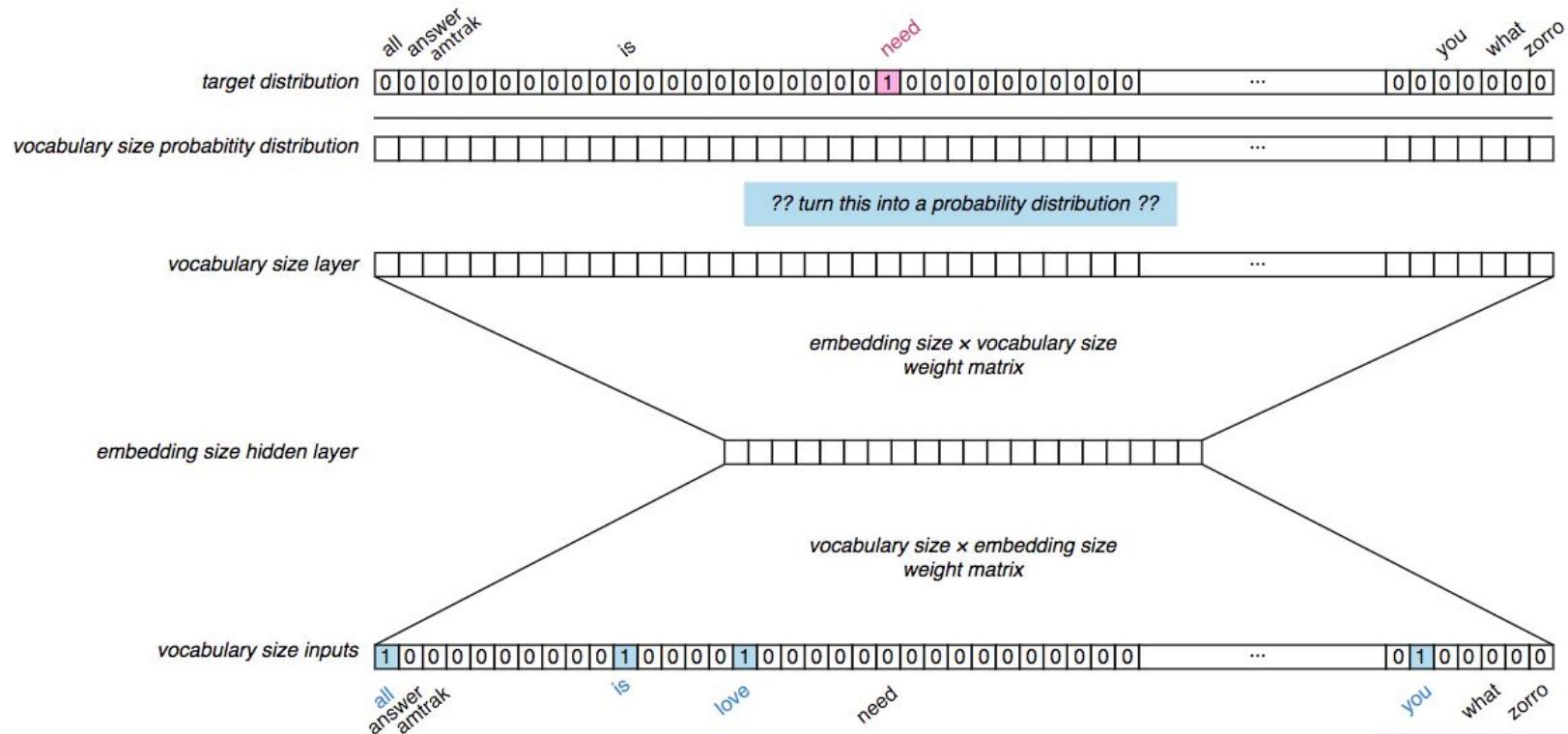


- Pre-trained word vectors. This data is made available under the [Public Domain Dedication and License](http://www.opendatacommons.org/licenses/pddl/1.0/) v1.0 whose full text can be found at: <http://www.opendatacommons.org/licenses/pddl/1.0/>.
  - [Wikipedia 2014 + Gigaword 5](#) (6B tokens, 400K vocab, uncased, 50d, 100d, 200d, & 300d vectors, 822 MB download): [glove.6B.zip](#)
  - Common Crawl (42B tokens, 1.9M vocab, uncased, 300d vectors, 1.75 GB download): [glove.42B.300d.zip](#)
  - Common Crawl (840B tokens, 2.2M vocab, cased, 300d vectors, 2.03 GB download): [glove.840B.300d.zip](#)
  - Twitter (2B tweets, 27B tokens, 1.2M vocab, uncased, 25d, 50d, 100d, & 200d vectors, 1.42 GB download): [glove.twitter.27B.zip](#)

# Embeddings

word2vec/GloVe: the vector of a word should be similar to the vectors of its neighbouring words

## Context-free: one embedding per word



... representation in a new space that should preserve the relationships between items of the original representation.

# Embeddings

Dense vector representation.

Low-dimensional.

Learnt from data.

Distributed representation most often refers to learnt embeddings.

In today's NLP literature, the default encoding.

Motivated by the **distributional hypothesis.**

This is an old idea! LSI was one of the first practical 'implementations' of it (1988). Did not work well.



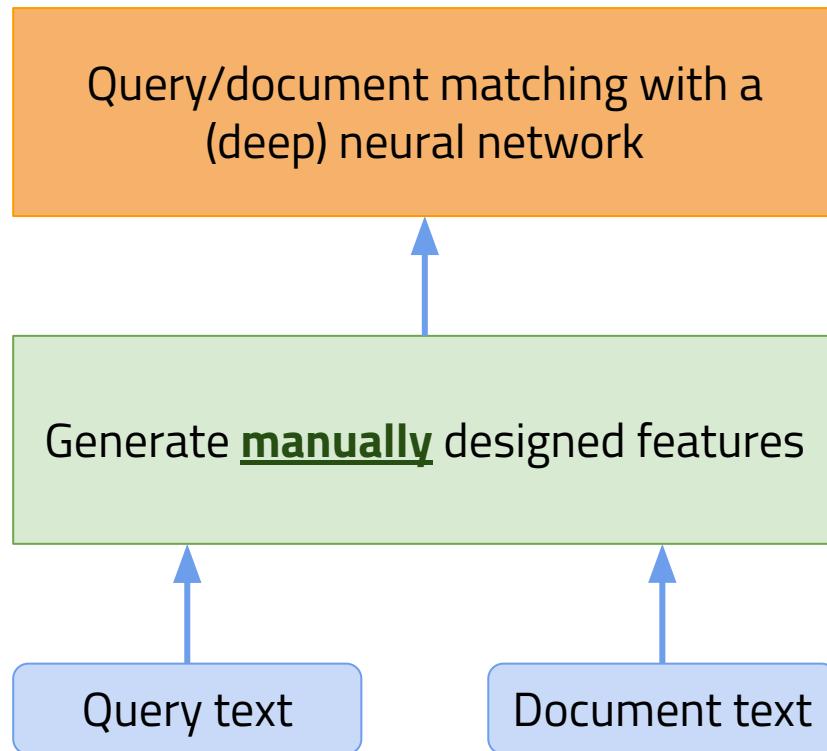
# Neural IR architectures

Categorized according to query representation, document representation and relevance estimation.

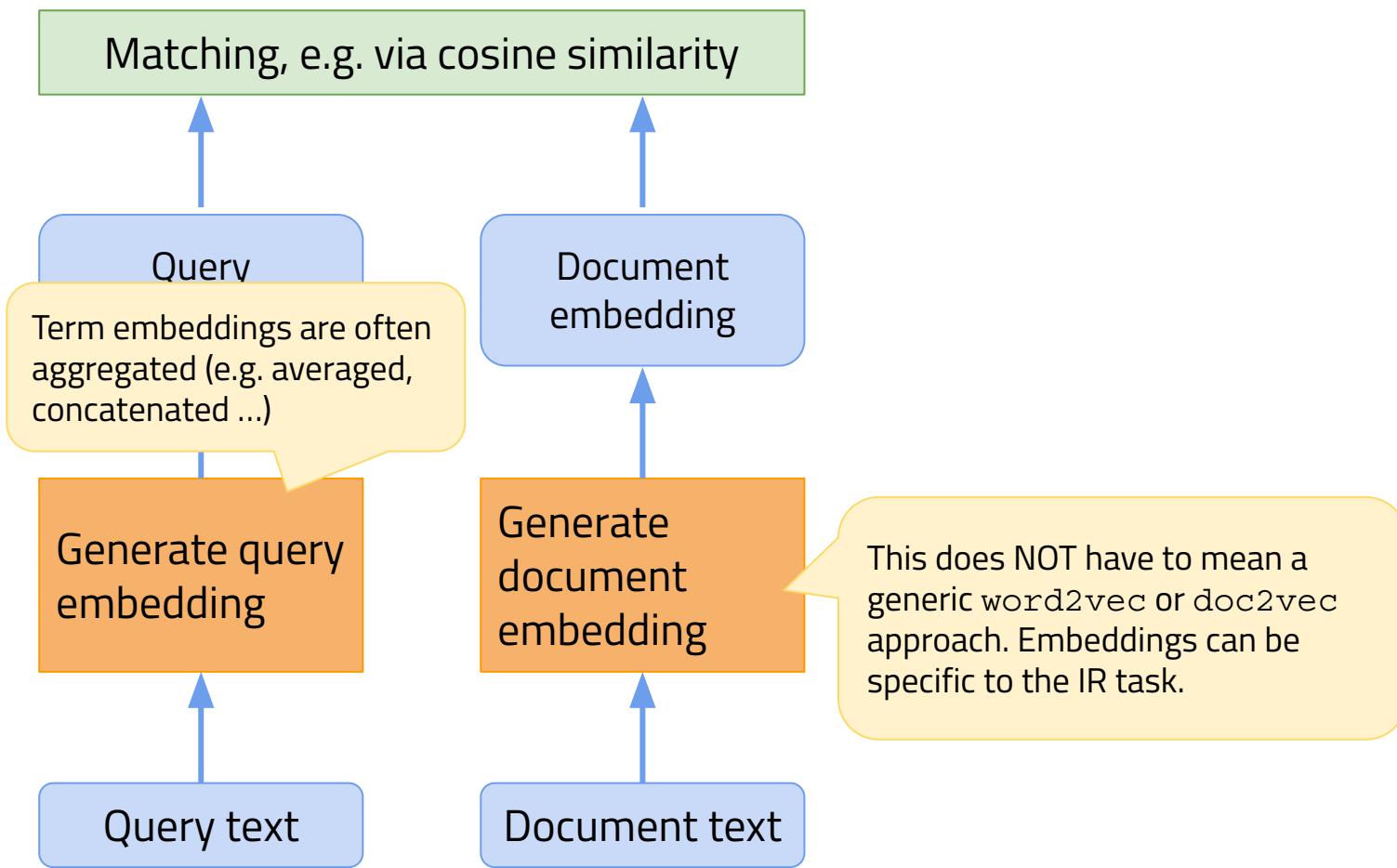
# A good retrieval system should ...

1. Have **semantic understanding** and enable **exact term matching**
2. Be **robust** to rare **inputs**: remember the long tail
3. Be **robust** to **corpus variance**
4. Be **robust** to **variable length input**

# Let's start with L2R's architecture



# Representation focused models



	Models	NDCG@1
1	BM25	0,305
5	DSSM	0,320
6	C-DSSM	<b>0,342</b>

# Example: C-DSSM

## Convolutional Deep Structured Semantic Model

Semantic layer:  $y$

Affine projection matrix:  $W_s$

Max pooling layer:  $v$

Max pooling operation

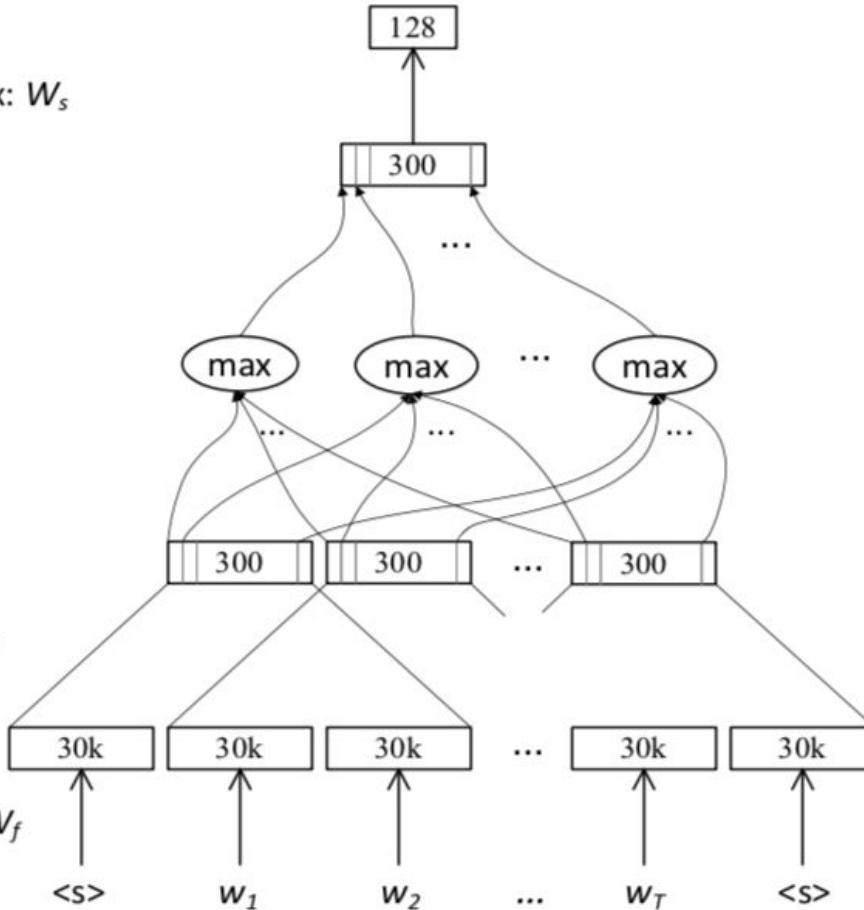
Convolutional layer:  $h_t$

Convolution matrix:  $W_c$

Word hashing layer:  $f_t$

Word hashing matrix:  $W_f$

Word sequence:  $x_t$

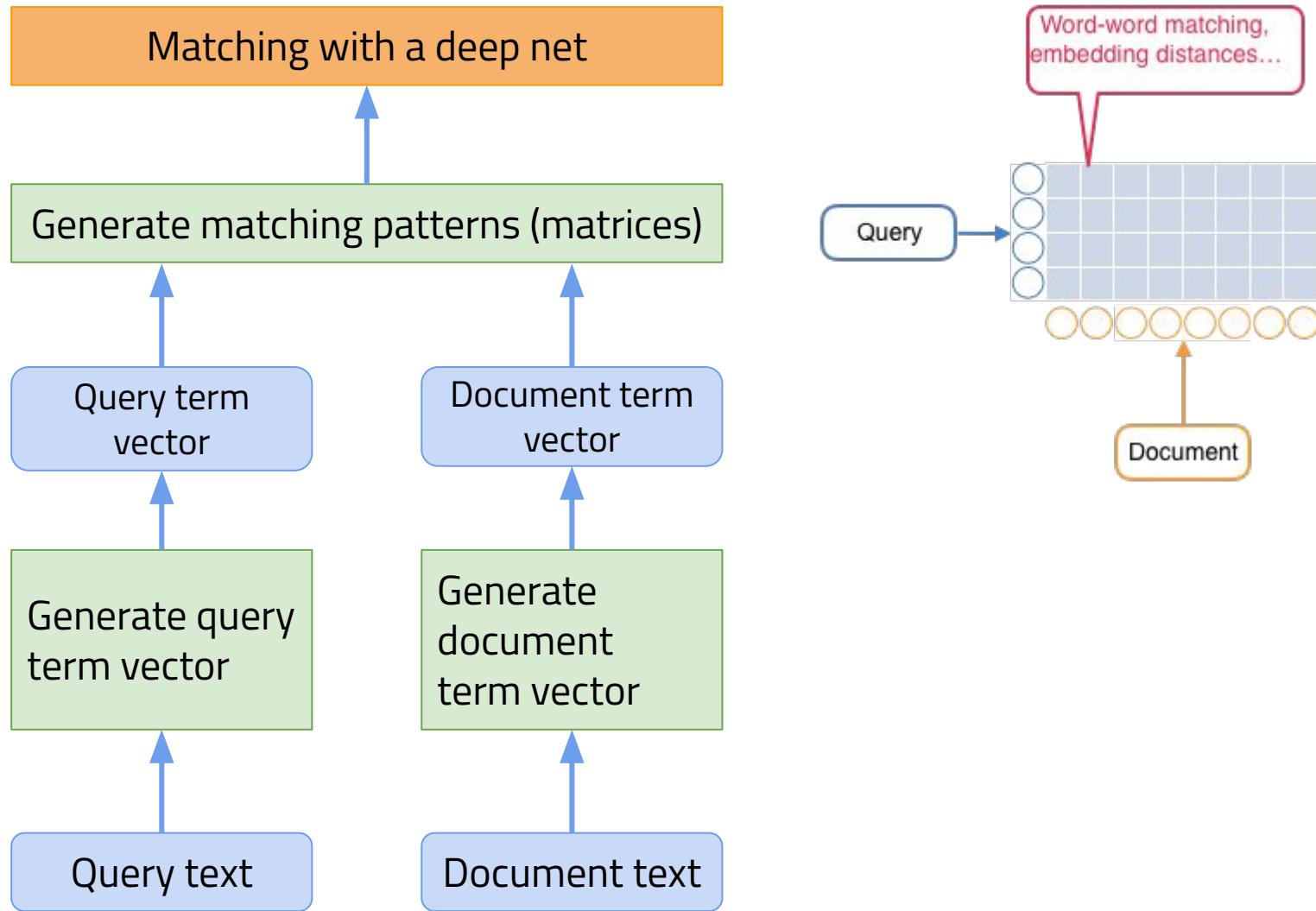


(30M query-clicked titles instances for training; document titles only)

Query-doc similarity based on cosine similarity

char trigram representation

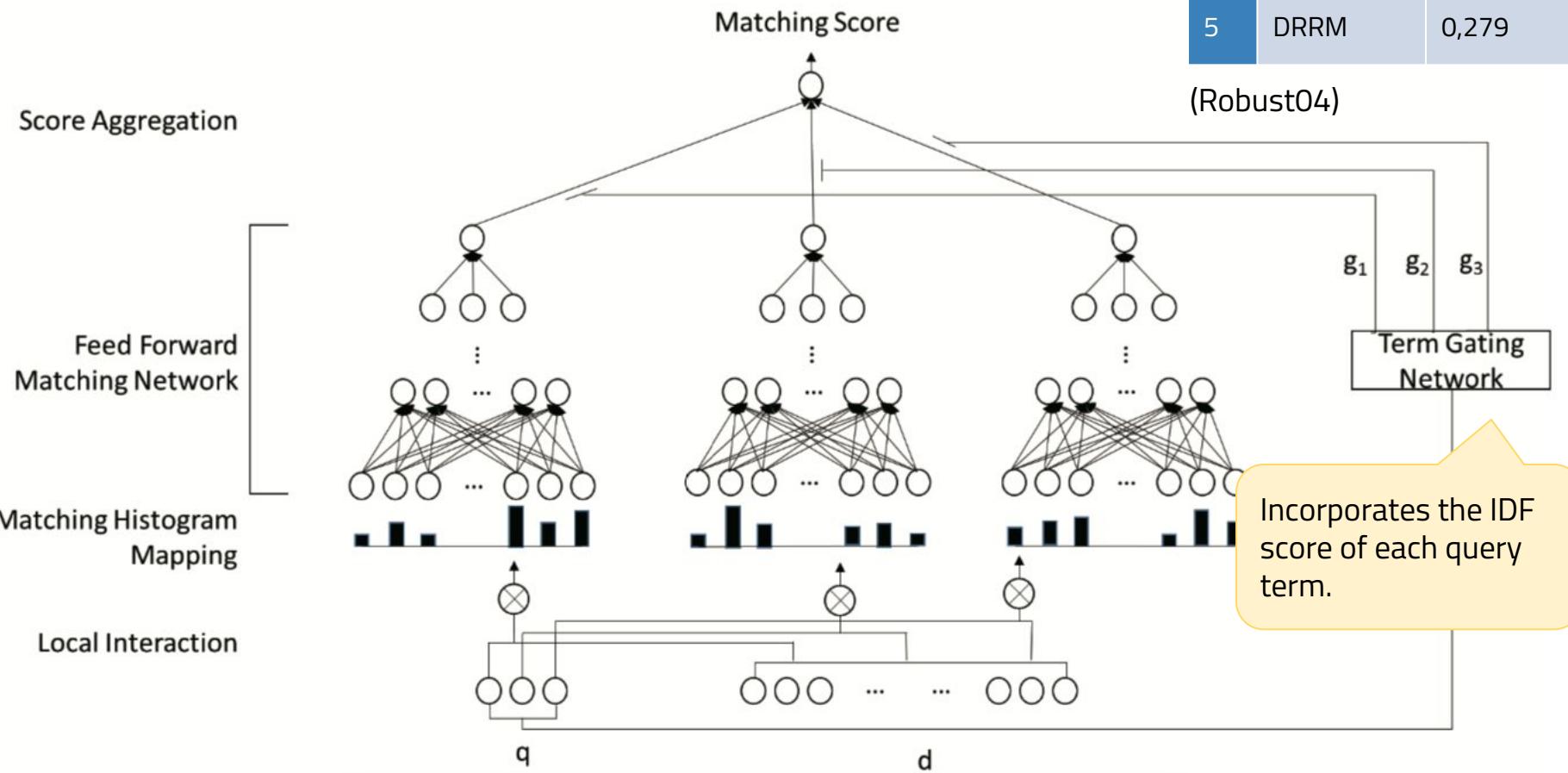
# Interaction focused models



	Models	MAP
1	QL	0,253
2	BM25	0,255
3	CDSSM	0,067
4	ARC-II	0,067
5	DRRM	0,279

# Example: DRMM

Deep Relevance Matching Model



joint training

# Hybrid example: Duet

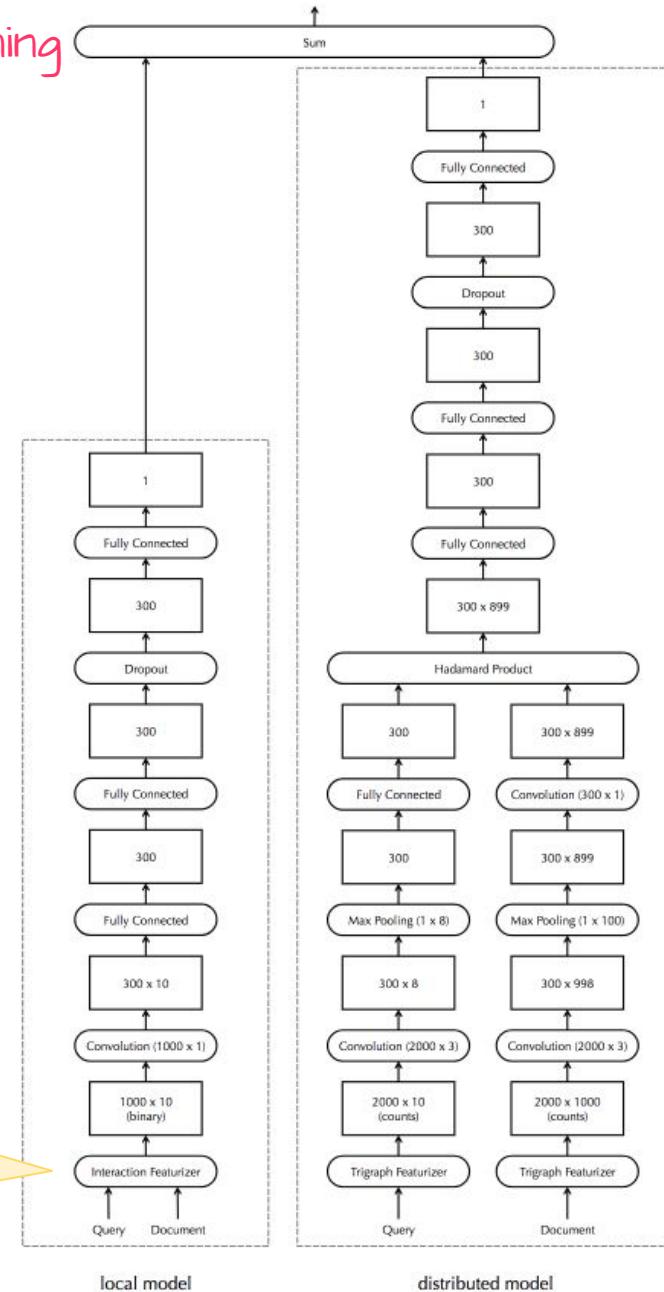
$$f(\mathbf{Q}, \mathbf{D}) = f_{local}(\mathbf{Q}, \mathbf{D}) + f_{distr}(\mathbf{Q}, \mathbf{D})$$

**Fixed** input length: first 10 terms per query and first 1000 terms per document

Local model: one-hot encoded input

Distributed model: n-grams as input

Patterns  
of term  
matches



	Models	NDCG@1
1	QL	0,246
2	CDSSM	0,273
3	Duet local	0,246
4	Duet distr.	0,286
5	Duet	0,322

# Hybrid example: Duet

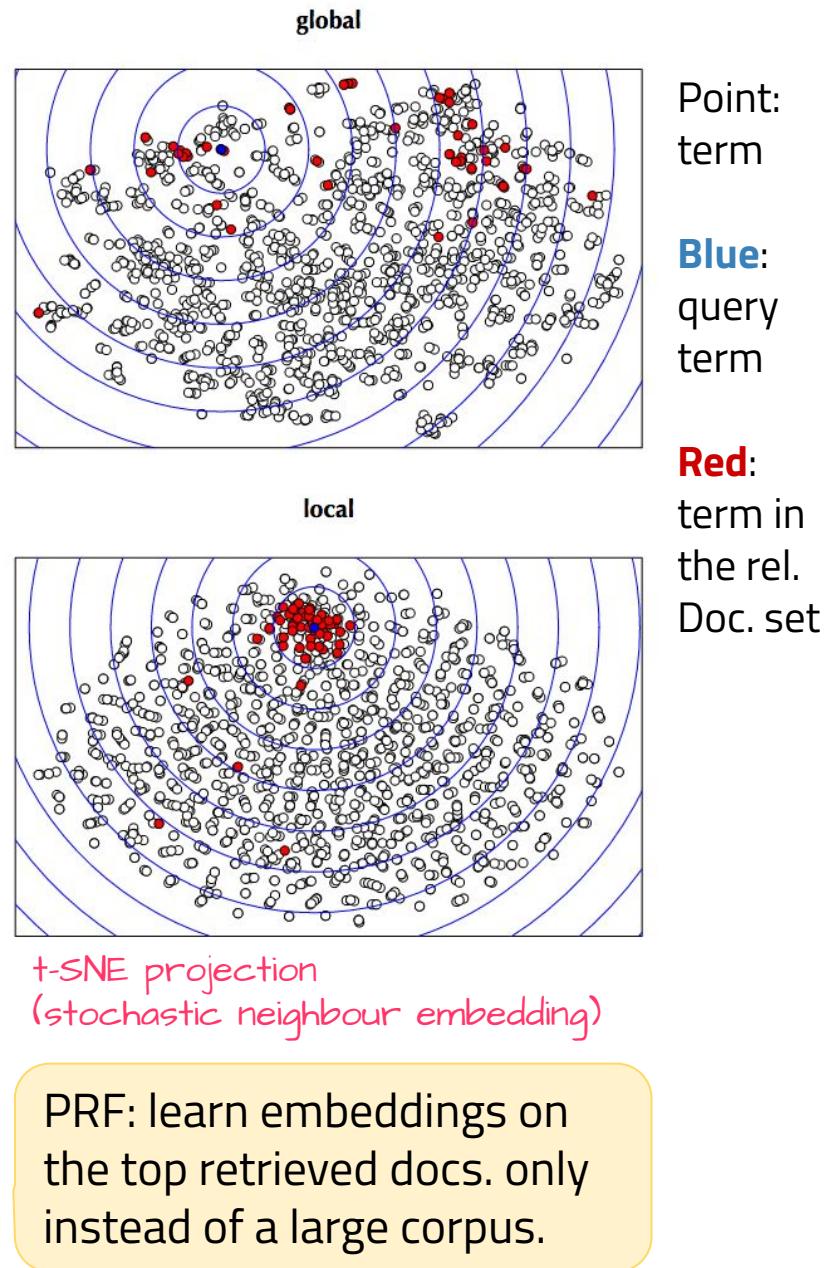
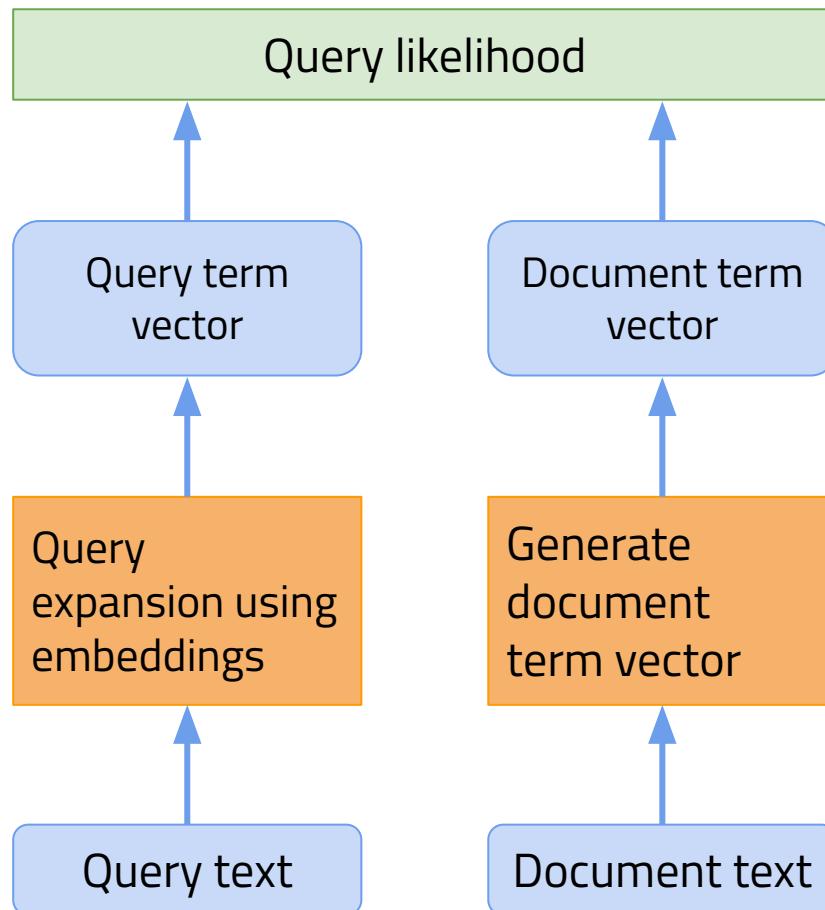
Training data: **200K queries** sampled from Bing's search logs.

Test data: **8K queries**, with approx. 25 documents per query  
**(14%** of queries also occurred in the training set)

Four negative documents sampled per relevant document.

**Domain knowledge is key:** *"For training, we discarded all documents rated as perfect because a large portion of them fall under the navigational intent, which can be better satisfied by historical click based ranking signals."*

# Query expansion



# Insights

Many specific architectures have been proposed,  
little work on **automatic architecture search**

**Interaction-focused techniques** tend to  
outperform representation-based models [17]

**Full ranking** instead of re-ranking remains  
challenging (**sparse indices**)

**Relevance matching** != semantic matching



**Reproducibility** is a big issue (many  
Implementation details to get right)

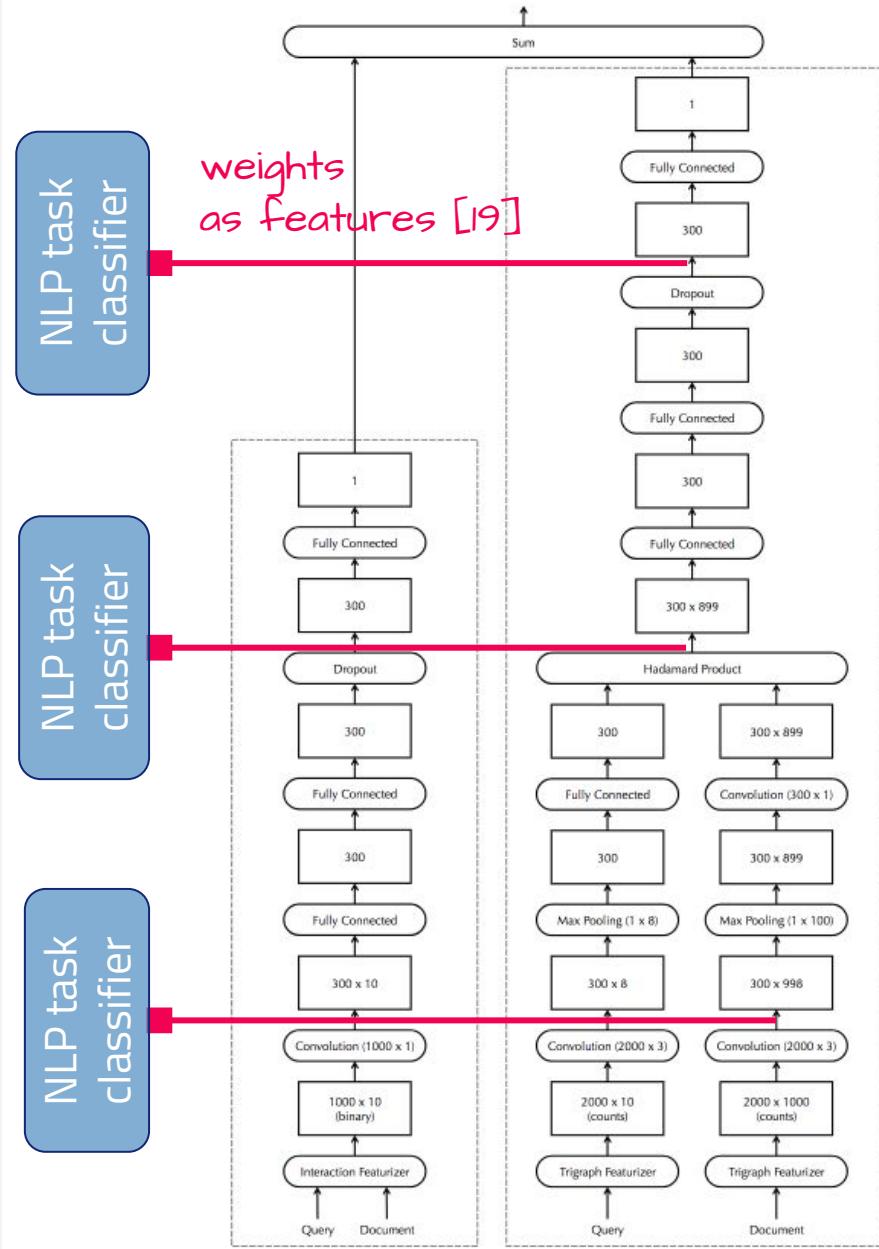
We need a lot of **training data** - how to get it?

**Model analysis** is in its infancy



[17] Nie, Yifan, Yanling Li, and Jian-Yun Nie. "Empirical study of multi-level convolution models for ir based on representations and interactions." ICTIR. ACM, 2018.

# Model analysis

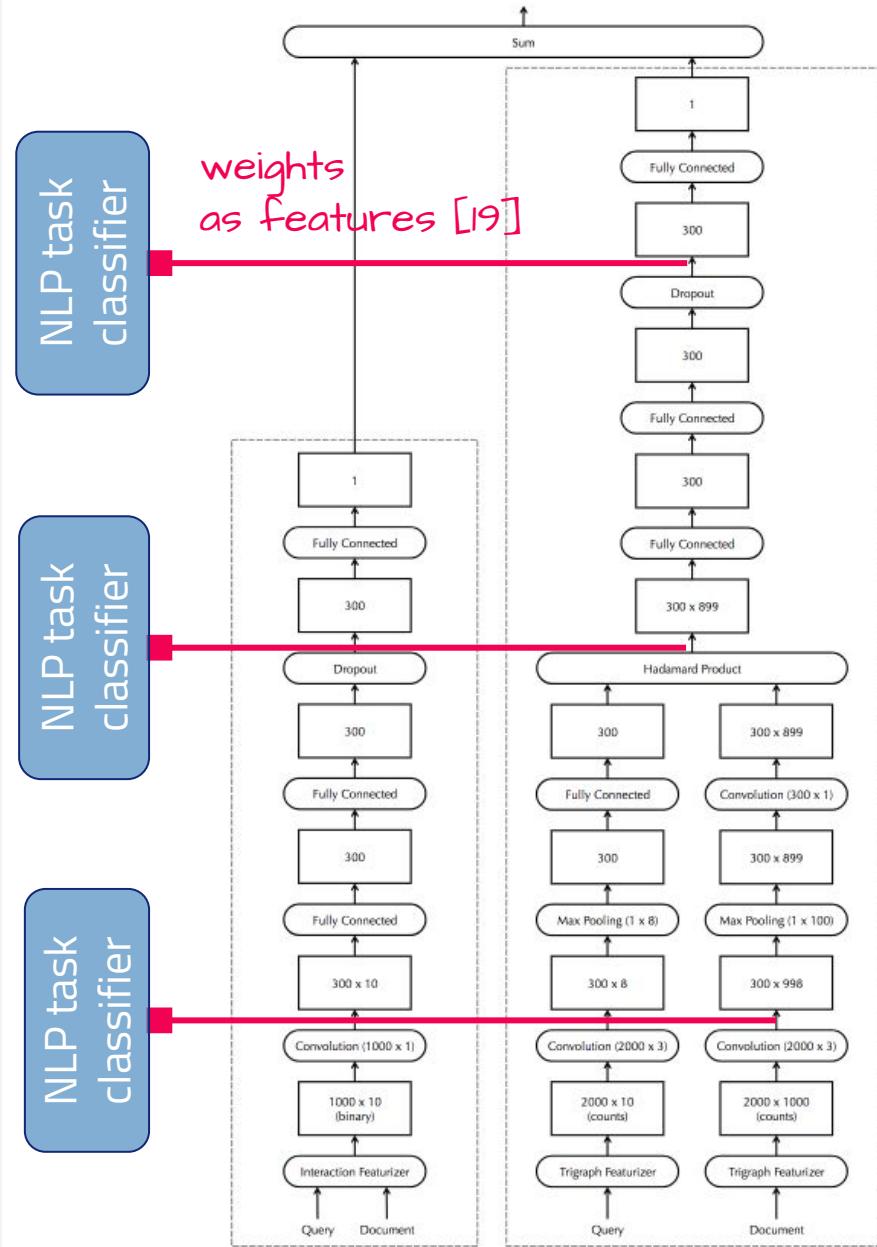


- [18] Rennings, Daniël, Felipe Moraes, and Claudia Hauff. "An Axiomatic Approach to Diagnosing Neural IR Models." ECIR, 2019.
- [19] Cohen, Daniel, Brendan O'Connor, and W. Bruce Croft. "Understanding the representational power of neural retrieval models using NLP tasks." ICTIR. ACM, 2018.

# Model analysis

Diagnostic approach [18]:

- Create diagnostic dataset to fulfill an **axiom**; evaluate to what extent neural models realize the patterns
- TFC1: given a single-term query  $w$  and two equally long documents, the RSV of the document with a higher freq. of  $w$  should be higher
- LNC2: If a document is replicated  $k$  times, its RSV should not be lower than that of its un-replicated variant
- Neural models fare poorly on LNC2



[18] Rennings, Daniël, Felipe Moraes, and Claudia Hauff. "An Axiomatic Approach to Diagnosing Neural IR Models." ECIR, 2019.

[19] Cohen, Daniel, Brendan O'Connor, and W. Bruce Croft. "Understanding the representational power of neural retrieval models using NLP tasks." ICTIR. ACM, 2018.

Issue across IR, not specific to neural models.  
Ongoing efforts (e.g. OSIRRC 2019).

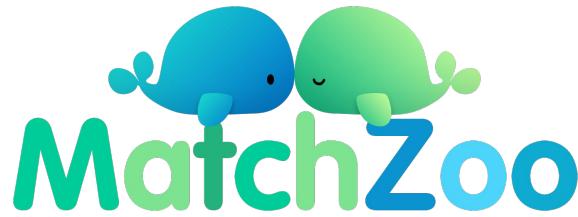
## Baselines: well-tuned?

Condition	AP	P20
QL	0.2499	0.3556
QL + RM3	0.2865	0.3773
Neural <sub>1</sub>	0.2815	0.3752
Neural <sub>2</sub>	0.2801	0.3764
Neural <sub>3</sub>	0.2856	0.3766
Neural <sub>3</sub> '	0.2971	0.3948
Anserini: QL	0.2496	0.3543
Anserini: BM25	0.2526	0.3604
Anserini: BM25 + RM3 (independent)	0.2954	0.3885
Anserini: BM25 + RM3 (joint)	0.2973	0.3871

(Robust04)

[20] Lin, Jimmy. "The neural hype and comparisons against weak baselines." ACM SIGIR Forum.  
Vol. 52. No. 1. ACM, 2019.

OSIRRC @SIGIR 2019: <https://osirrc.github.io/osirrc2019/>



<https://github.com/NTMC-Community/MatchZoo>

Used by ▾

3

Watch ▾

157

Unstar

2,500

Fork

680

DRMM

MatchPyramid

ARC-I/-II

aNMM

DSSM

C-DSSM

Duet

...

Thanks @bwanglzu for pointing out the differences of the implementations in MatchZoo with the descriptions in some papers. Yes, this is possible. We tried to implement the most important components/novel parts of these neural models. But it is still possible that there are some differences between our current implementation details with some details described in the paper. For some critical differences, we will fix them in the next version of MatchZoo. But for some differences like "dropout", I think it is fine to keep it. You can adjust the dropout rate to control the network as you want. What do you think about it ? @faneshion @pl8787 @bwanglzu

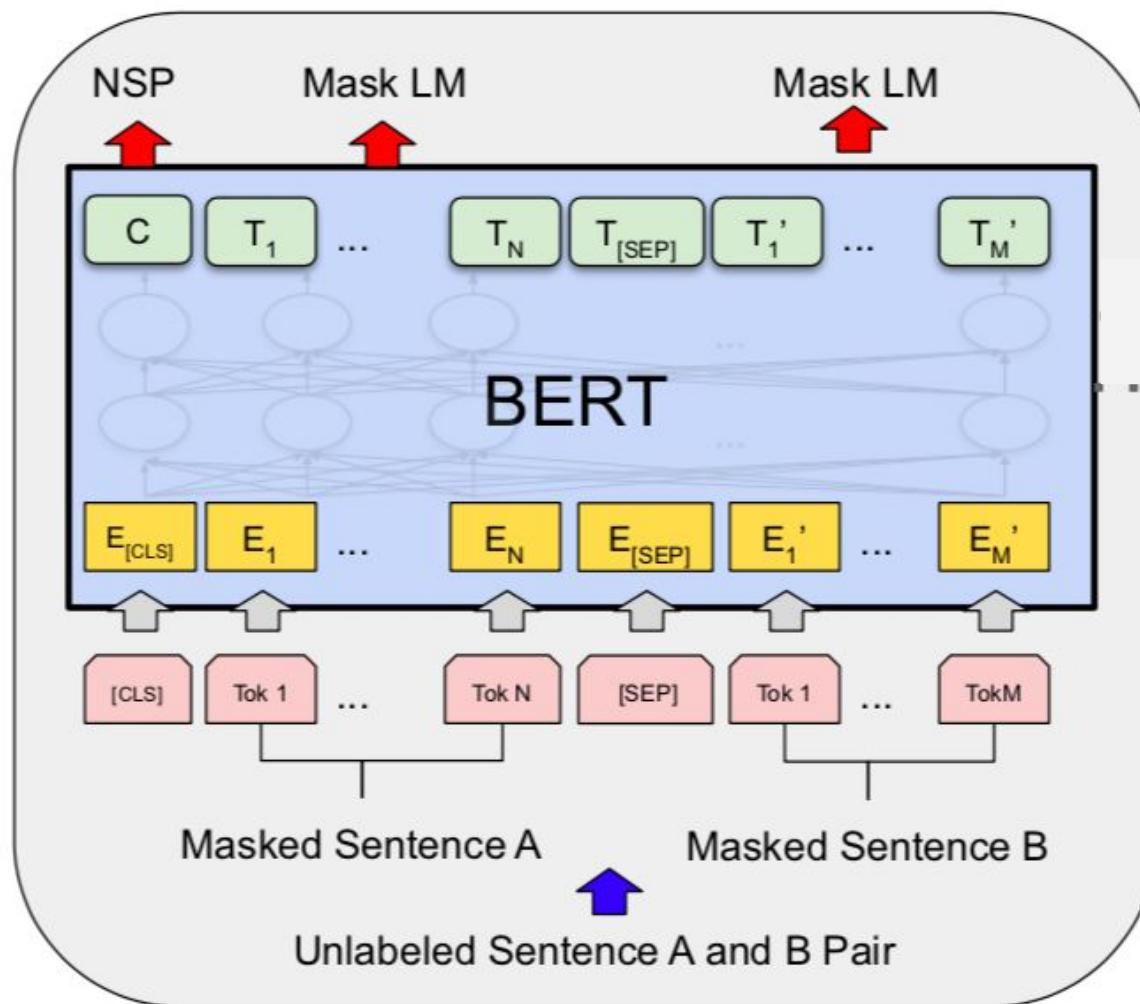
<https://github.com/NTMC-Community/MatchZoo/issues/99>

# Finally ... BERT

Bidirectional (contextual) Encoder Representations  
from Intransformers (a popular attention model)



# BERT: pre-training



Pre-training based on:

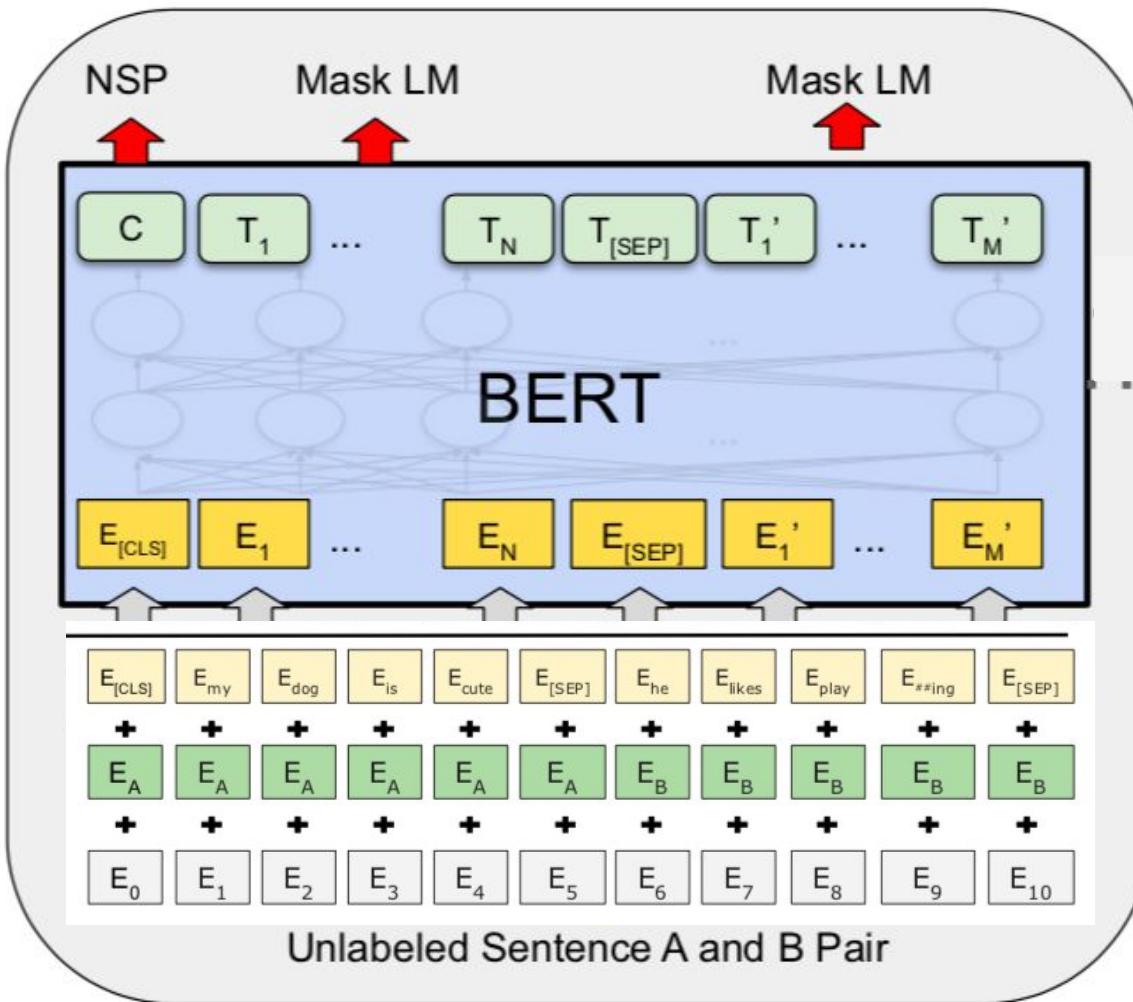
1. Masked language model (15% tokens)
2. Next sentence prediction

(i.e. *self-supervised*)

**340M** parameters

Finding: BERT reduces the need for task-specific architectures (fine-tuning of the pre-trained model is enough)

# BERT: pre-training



Pre-training based on:

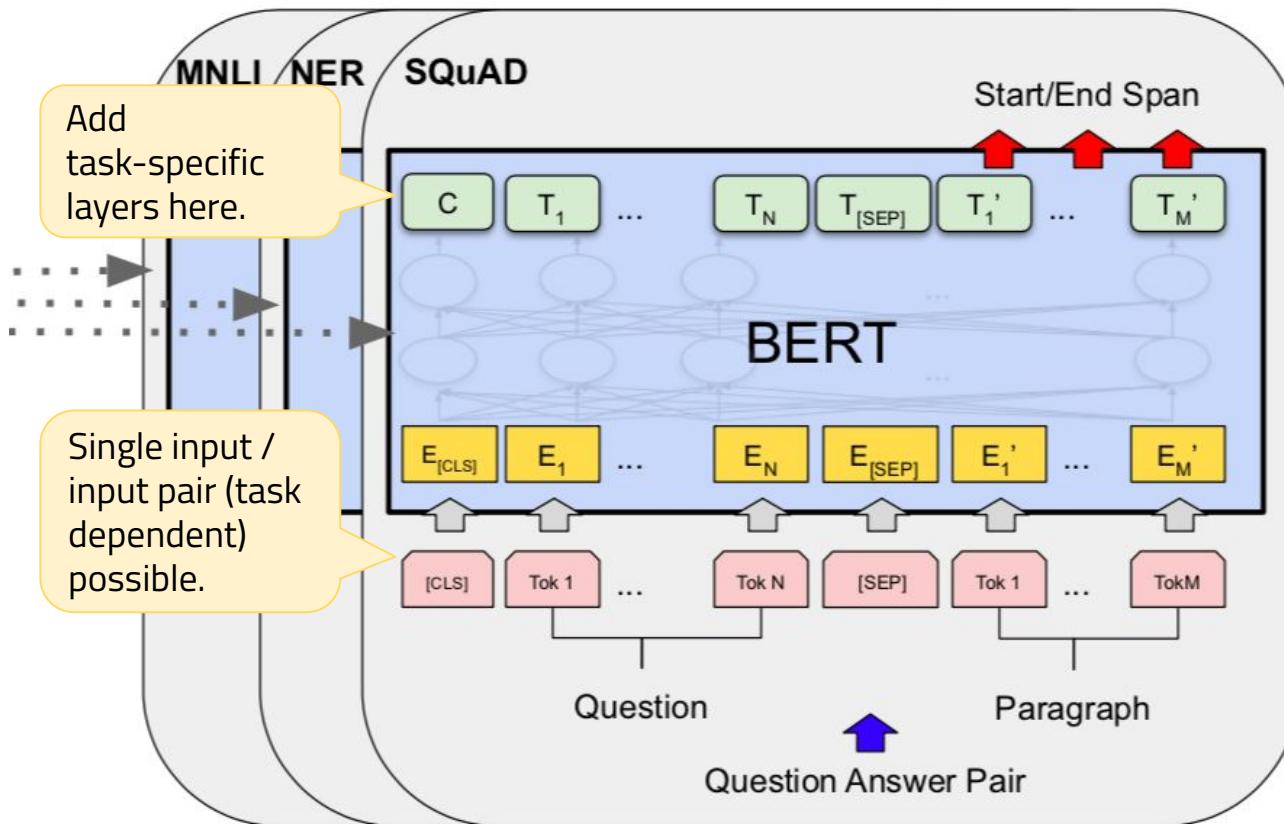
1. Masked language model (15% tokens)
2. Next sentence prediction

(i.e. *self-supervised*)

**340M** parameters

Finding: BERT reduces the need for task-specific architectures (fine-tuning of the pre-trained model is enough)

# BERT: fine-tuning



1. Initialized with pre-trained parameters
2. Adapt the final layer in a task-specific manner.
3. Fine-tune with labelled data specific to your task

# Passage re-ranking with BERT

MSMarco (1M+ queries):

1. Retrieve top- $k$  passages with BM25 (computationally inexpensive)
2. Rerank top- $k$  passages based on fine-tuned BERT

Fine-tuning:

- Query: "Sentence A" (max. 64 tokens)
- Passage: "Sentence B" (query+passage+[SEP] max. 512 tokens)
- Output: probability of relevance
- Loss function: cross-entropy loss

# Passage re-ranking with BERT

MSMarco (1M+ q)

1. Retrieve top- $k$  passages (inexpensive)

$$\text{Relevance} = \mathbf{w}^T * \mathbf{C}_k$$

Trained with cross-entropy loss

2. Rerank top- $k$  passages with BERT

BERT

Fine-tuning:

- Query: "Sentence A"
- Passage: "Sentence B"
- Output: probability
- Loss function

$E_{CLS}$

$E_1$

$E_n$

$E_{SEP}$

$E_1'$

$E_M'$

CLS

$W^1$

$W^N$

SEP

$W^1$

$W^M$

query

passage

# Passage re-ranking with BERT

MSMarco (1M+ queries):

1. Retrieve top- $k$  passages with BM25 (computationally inexpensive)
2. Rerank top- $k$  passages based on fine-tuned BERT

Fine-tuning:

- Query: "Sentence A" (max. 64 tokens)
- Passage: "Sentence B" (query+passage+[SEP] max. 512 tokens)
- Output: probability of relevance
- Loss function: cross-entropy loss

Result: **27%** relative improvement in MRR over previous SOTA!

# BERT for ad-hoc retrieval (newswire)

Insight: document relevance can be approximated by the “best” sentence or paragraph in a document

Idea: combine **document retrieval score** with **sentence-level BERT scores** (**sentence-level BERT scores**); few hyperparameters

$$\text{Score}_d = \underline{a} \cdot \underline{S_{\text{doc}}} + (1 - \underline{a}) \cdot \sum_{i=1}^n w_i \cdot \underline{S_i}$$

Fine-tune on datasets with sentence-level qrels, e.g. TREC microblog datasets (instead of the target corpus Robust04)

# BERT for ad-hoc retrieval (newswire)

Insight: document relevance can be approximated by the “best” sentence or paragraph in a document

Idea: combine Robust04

scores (sent)

Score <sub>d</sub>				
	BM25+RM3	0.3033	0.3974	
	1S: BERT FT(QA)	0.3102	0.4068	
	2S: BERT FT(QA)	0.3090	0.4064	
	3S: BERT FT(QA)	0.3090	0.4064	
	1S: BERT FT(Microblog)	0.3266	0.4245	
Fine-tune on datasets (ins	2S: BERT FT(Microblog)	<b>0.3278</b>	0.4267	
	3S: BERT FT(Microblog)	<b>0.3278</b>	<b>0.4287</b>	

# BERT is not the last word!

# XLNet just came out.

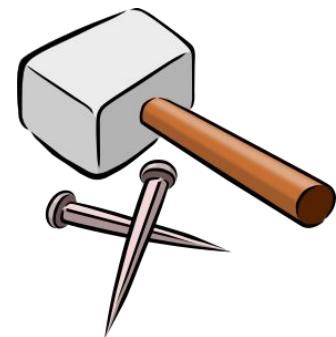
Very worthwhile tutorial on Writing code for NLP (and IR) research:  
<http://tiny.cc/pxes9y>

“

However, given the pace at which the area of deep learning is growing [...] we should be wary of the **combinatorial explosion of trying every model on every IR task.**

We should not disproportionately focus on maximizing quantitative improvements and in the process **neglect theoretical understanding and qualitative insights.** [...]

Neural models should not be the hammer that we try on every IR task, or we may risk reducing every IR task to a nail.



- ☒ Learning to rank
- ☒ Neural IR
- ☒ Research directions
- ☒ Pitfalls

✉ [c.hauff@tudelft.nl](mailto:c.hauff@tudelft.nl)  
🌐 [chauff.github.io](https://chauff.github.io)

