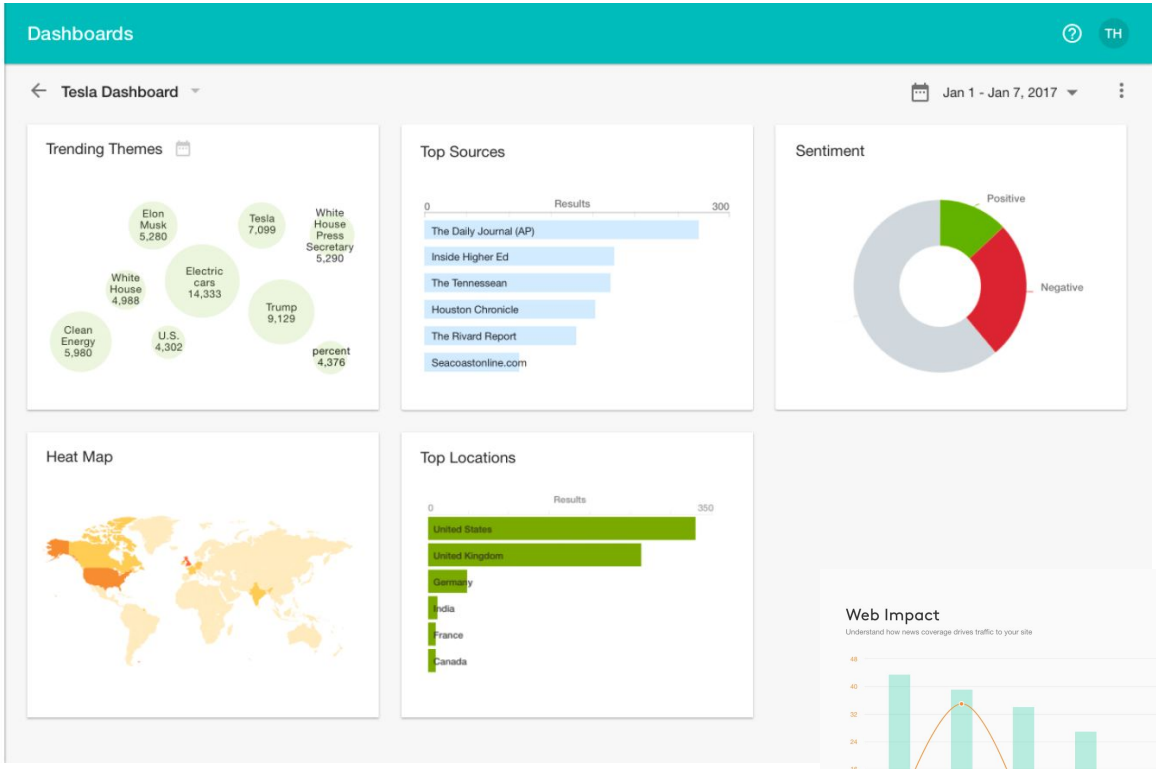
Predicting Resource Consumption in a Large-Scale Information Retrieval System

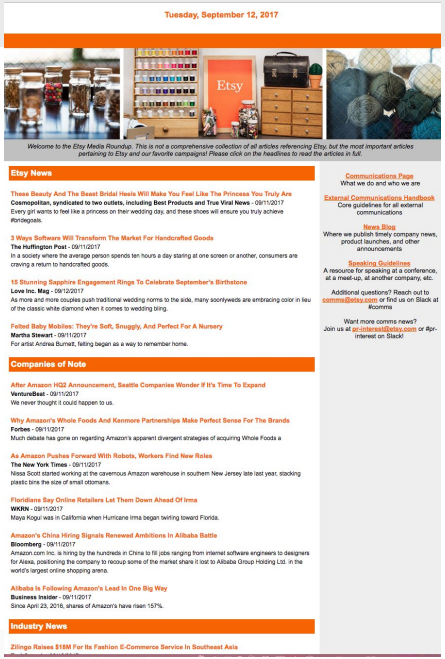
Some ideas on dealing with inherently noisy data in Machine Learning

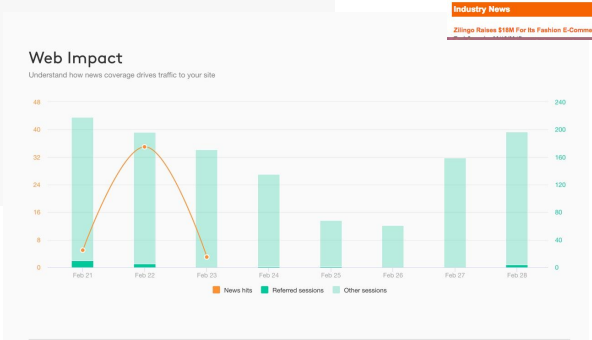
Presented by: David Burke

Meltwater

Who are Meltwater?







Meltwater

Large-Scale Information Retrieval

35B
AVAILABLE DOCUMENTS

2.2M
SEARCHES/DAY

75M
INCOMING DOCUMENTS/DAY

elastic

500
DATA NODES

99.96%
UPTIME/SERVICE LEVEL

168ms
MEDIAN RESPONSE TIME

Horace

Large-Scale Information Retrieval

35B
AVAILABLE DOCUMENTS

75M
INCOMING DOCUMENTS/DAY

2.2M
SEARCHES/DAY

elastic

500
DATA NODES

99.96%
UPTIME/SERVICE LEVEL

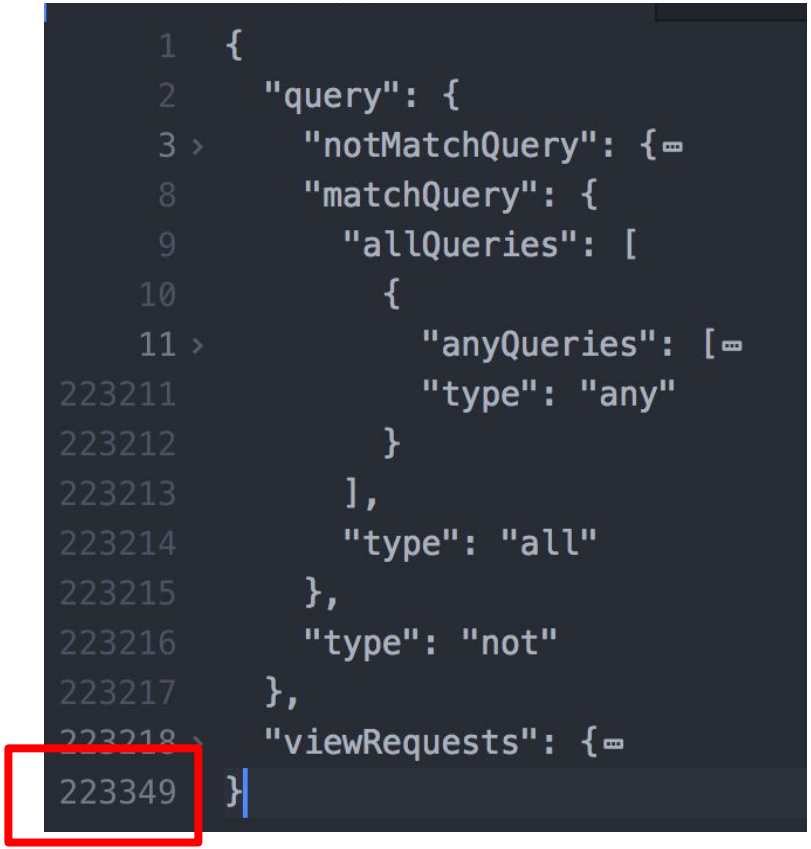
168ms
MEDIAN RESPONSE TIME

```
{
  "query": {
    "notMatchQuery": {
      "field": "body.content.text",
      "type": "term",
      "value": "Greek"
    },
    "matchQuery": {
      "allQueries": [
        {
          "field": "body.content.text",
          "type": "term",
          "value": "GAIA"
        },
        {
          "field": "body.content.text",
          "type": "term",
          "value": "Conference"
        },
        {
          "anyQueries": [
            {
              "field": "body.content.text",
              "type": "term",
              "value": "Gothenburg"
            },
            {
              "field": "body.content.text",
              "type": "term",
              "value": "Göteborg"
            }
```

Meltwater

Some fun queries

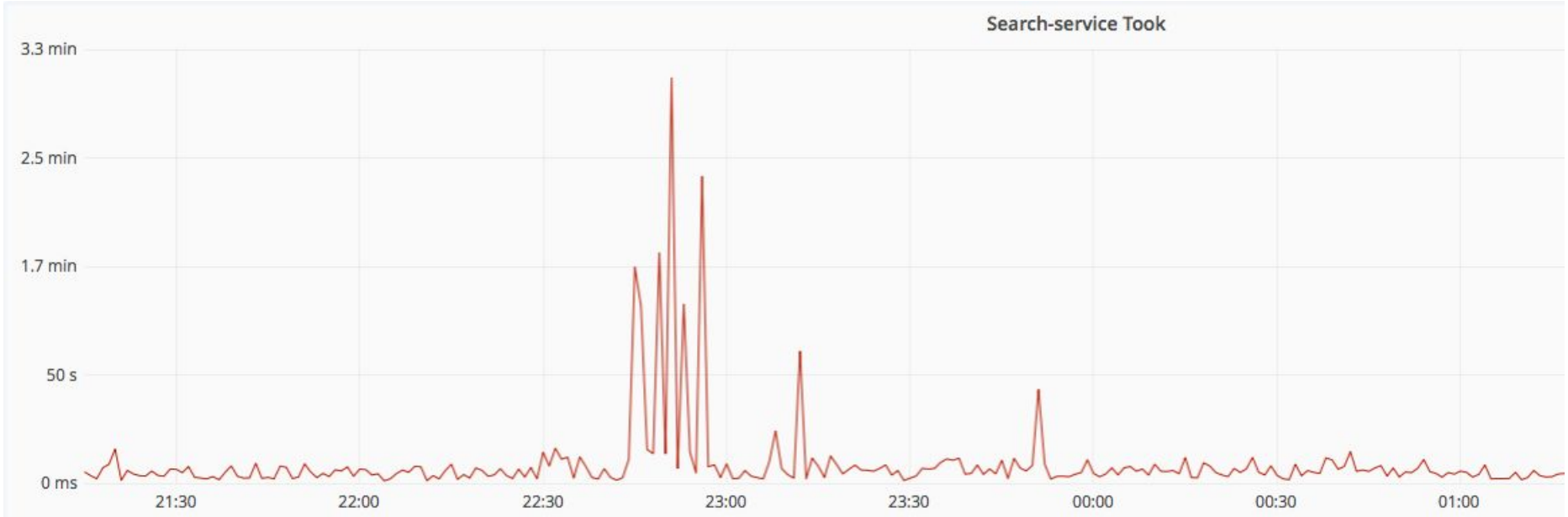
```
{
  "query": {
    "anyQueries": [
      {
        "field": "body.content.text",
        "type": "wildcard",
        "value": "a*"
      },
      {
        "field": "body.content.text",
        "type": "wildcard",
        "value": "b*"
      },
      {
        "field": "body.content.text",
        "type": "wildcard",
        "value": "c*"
      },
      {
        "field": "body.content.text",
        "type": "wildcard",
        "value": "d*"
      },
      ...
```



Meltwater

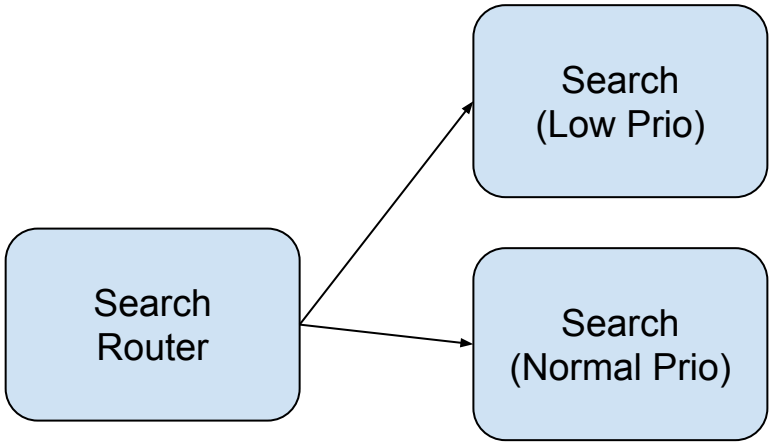
Search spikes

Search Times (average)

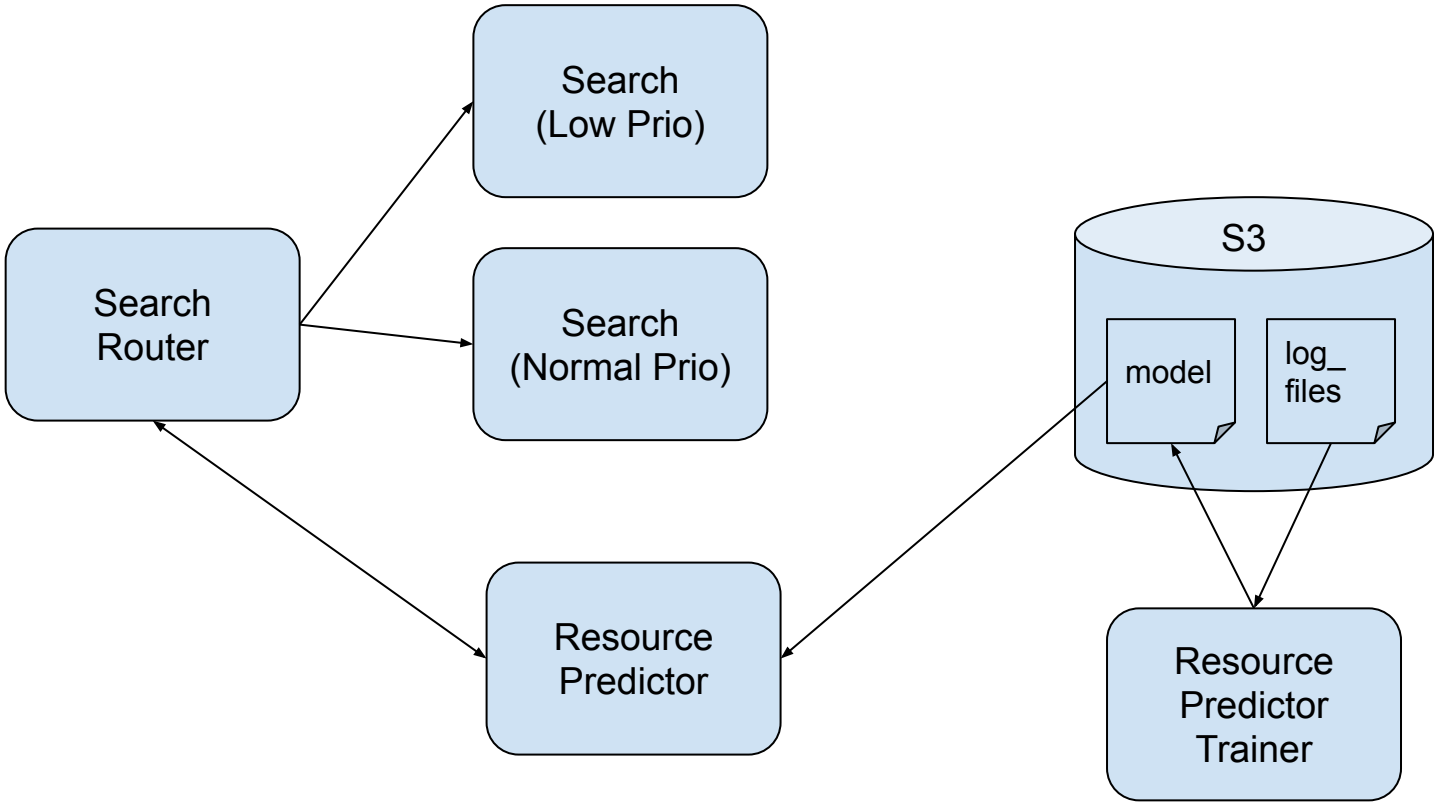


Meltwater

Query Resource Prediction for Routing

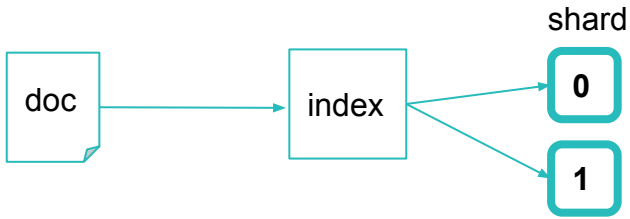


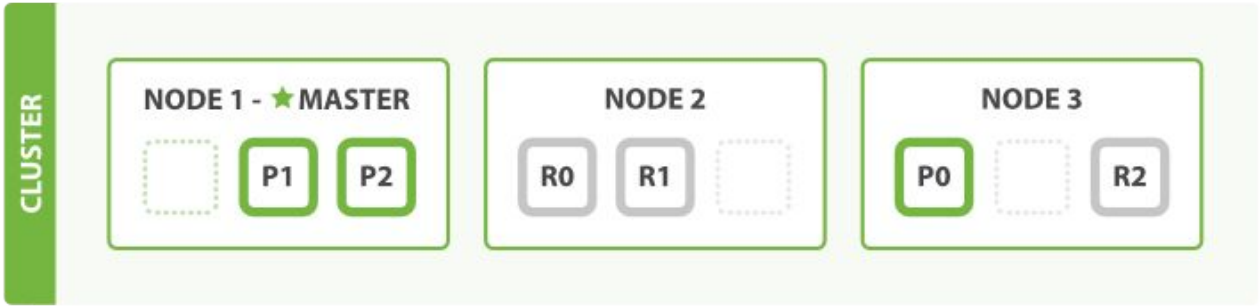
Query Resource Prediction for Routing



Elasticsearch

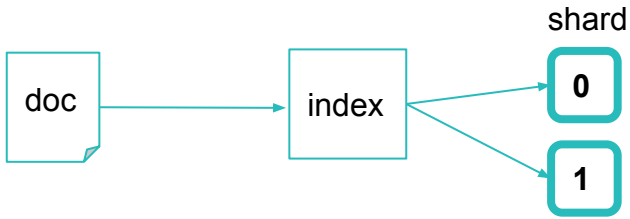
elasticsearch - scalable, distributed search



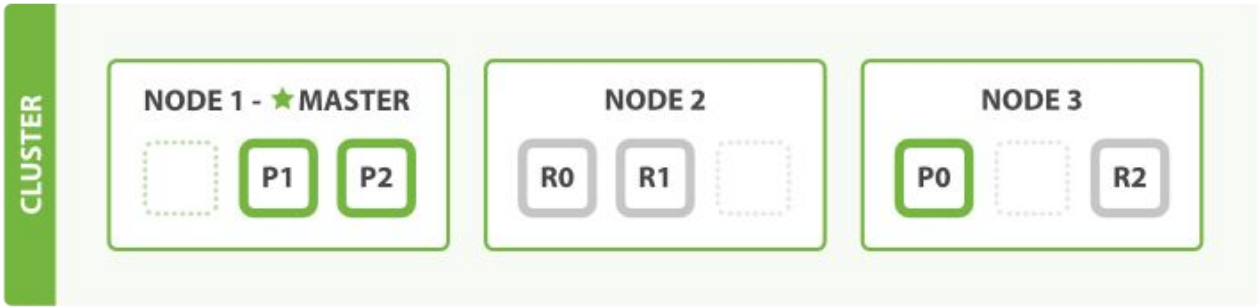


Elasticsearch

elasticsearch - scalable, distributed search



Resource consumption
≅query_time x # shards



Building a model

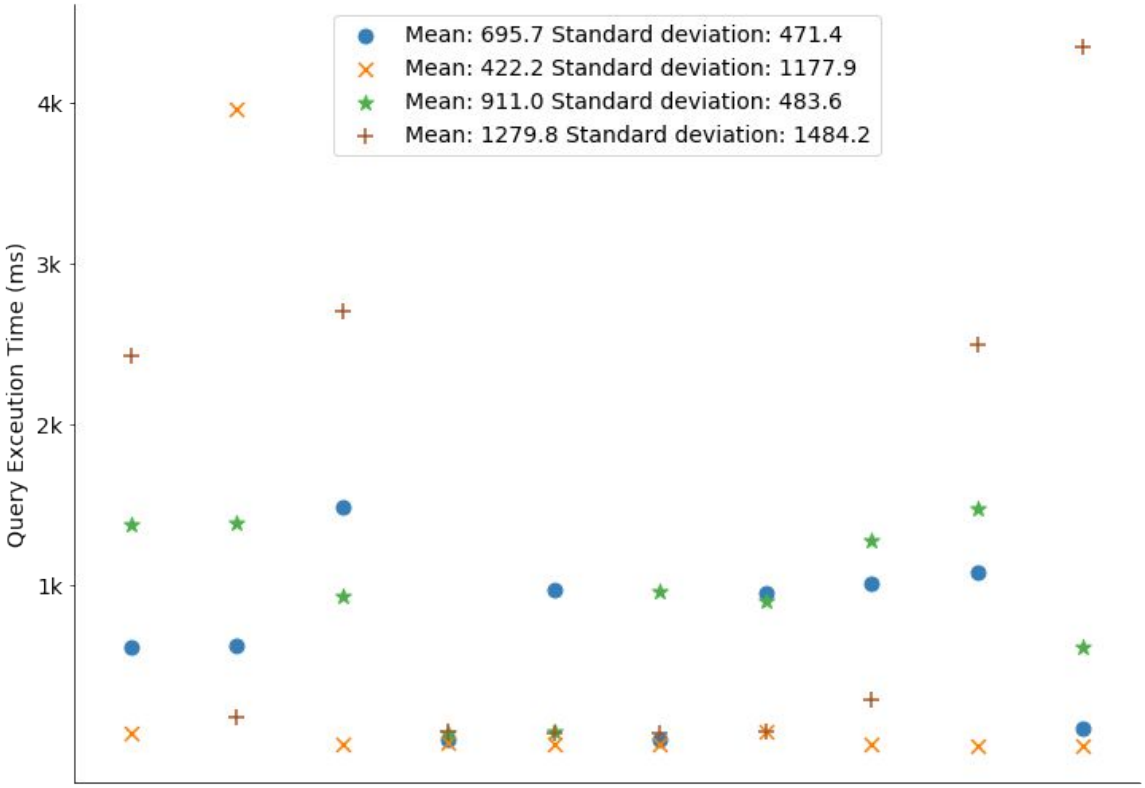
Query	$feature_1$	$feature_2$	...	$feature_n$	rc
q_1	f_{11}	f_{12}	...	f_{1n}	$\mathbf{r_1}$
q_2	f_{21}	f_{22}	...	f_{2n}	$\mathbf{r_2}$
...					
q_n	f_{n1}	f_{n2}	...	f_{nn}	$\mathbf{r_n}$

Building a model

Query	$feature_1$	$feature_2$	...	$feature_n$	rc
q_1	f_{11}	f_{12}	...	f_{1n}	r_1
q_2	f_{21}	f_{22}	...	f_{2n}	r_2
...					
q_n	f_{n1}	f_{n2}	...	f_{nn}	r_n

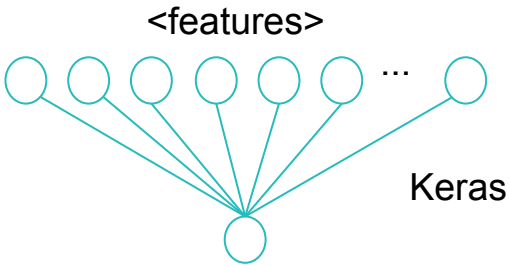
Building a model

Query	$feature_1$	$feature_2$	...	$feature_n$	rc
q_1	f_{11}	f_{12}	...	f_{1n}	r_1
q_2	f_{21}	f_{22}	...	f_{2n}	r_2
...					
q_n	f_{n1}	f_{n2}	...	f_{nn}	r_n



Meltwater

Building a model




```
Features =
{
        number_of_words,
        number_of_wildcards,
        number_of_ors,
        number_of_ands,
        number_of_nots,
        …
        etc.
}
```

Model Strategies

1. Dynamic/online noisy model

Building a model

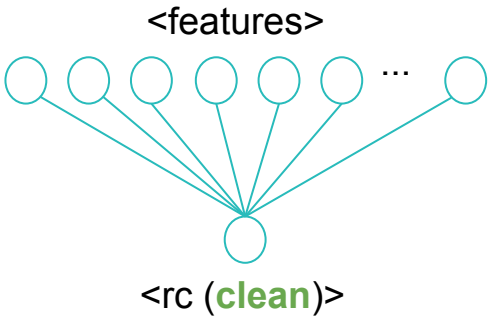
Query	$feature_1$	$feature_2$	...	$feature_n$	rc	rc_c
q_1	f_{11}	f_{12}	...	f_{1n}	$\mathbf{r_1}$	$\mathbf{r_{1c}}$
q_2	f_{21}	f_{22}	...	f_{2n}	$\mathbf{r_2}$	$\mathbf{r_{2c}}$
...						
q_n	f_{n1}	f_{n2}	...	f_{nn}	$\mathbf{r_n}$	$\mathbf{r_{nc}}$

Clean measurements

Model Strategies

1. Dynamic/online noisy model

Building a model



Model Strategies

1. Dynamic/online noisy model
2. Static/offline clean model

Building a model

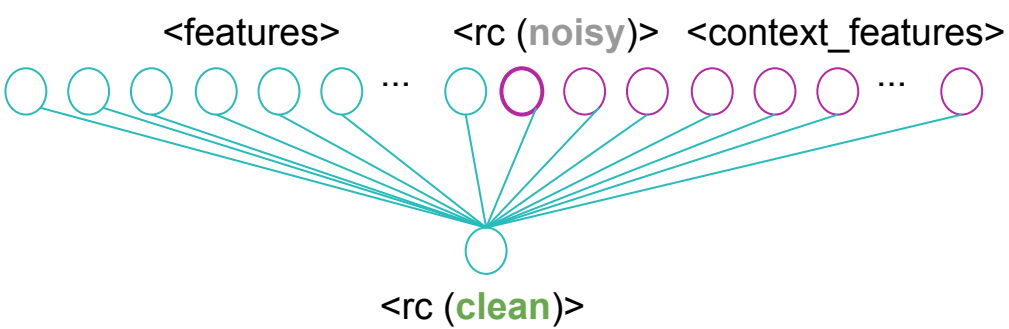
Query	$feature_1$	$feature_2$	...	$feature_n$	rc	rc_c	rc_d
q_1	f_{11}	f_{12}	...	f_{1n}	$\mathbf{r_1}$	$\mathbf{r_{1c}}$	$\mathbf{r_{1d}}$
q_2	f_{21}	f_{22}	...	f_{2n}	$\mathbf{r_2}$	$\mathbf{r_{2c}}$	$\mathbf{r_{1d}}$
...							
q_n	f_{n1}	f_{n2}	...	f_{nn}	$\mathbf{r_n}$	$\mathbf{r_{nc}}$	$\mathbf{r_{1d}}$

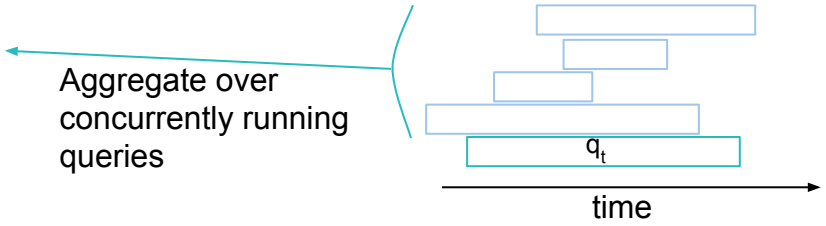
Denoised measurements

Model Strategies

1. Dynamic/online noisy model
2. Static/offline clean model

Building a model

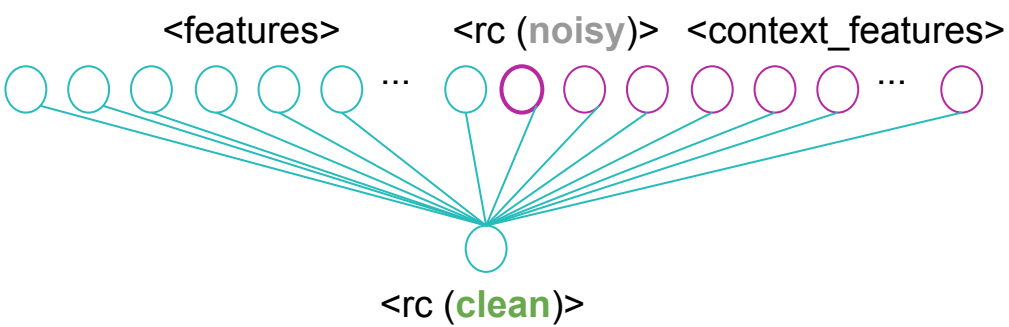


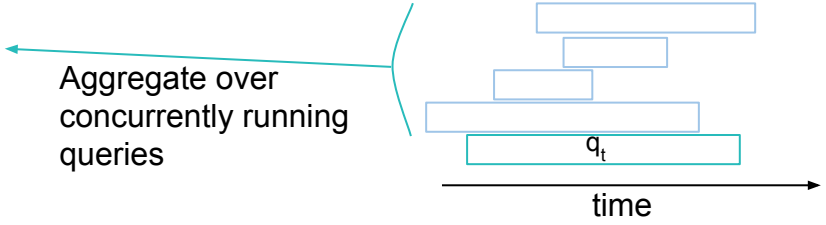


Model Strategies

1. Dynamic/online noisy model
2. Static/offline clean model
3. Dynamic/online denoised model
    - Denoise noisy measurements
    - Train on denoised measurements

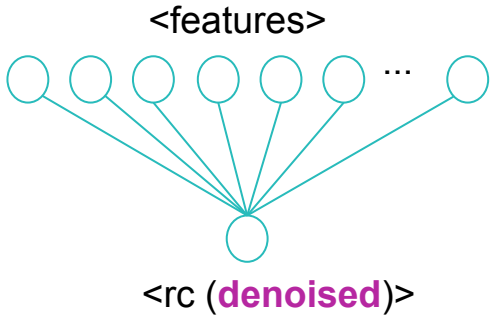
Building a model



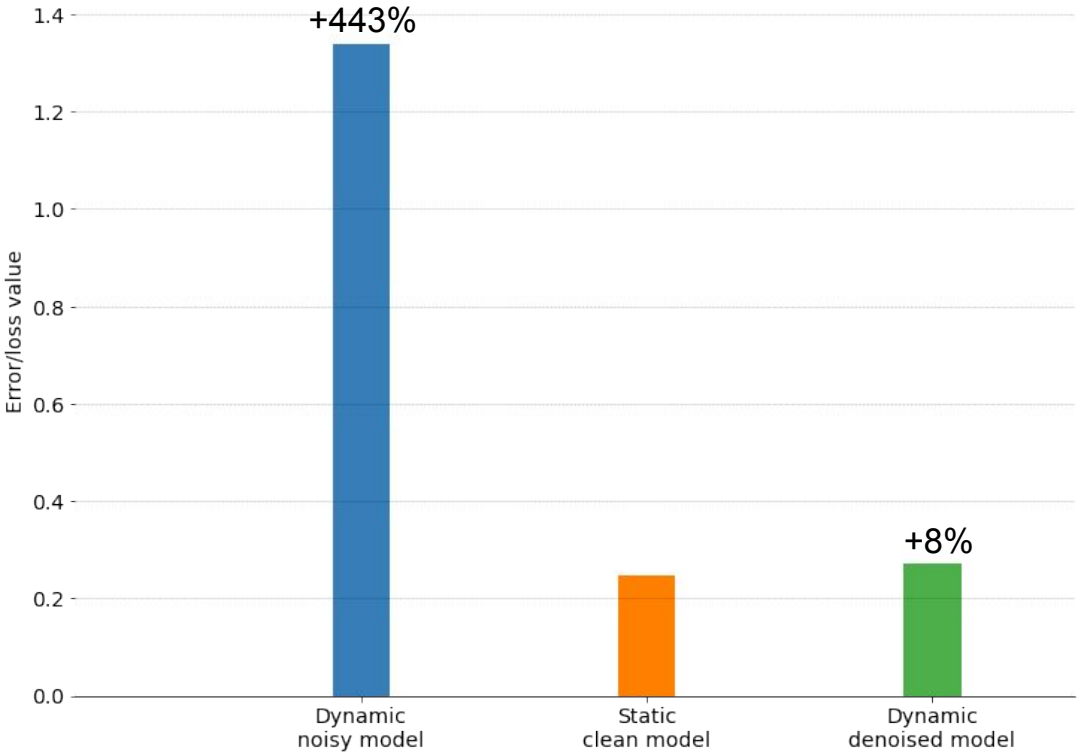


Model Strategies

1. Dynamic/online noisy model
2. Static/offline clean model
3. Dynamic/online denoised model
    - Denoise noisy measurements
    - Train on denoised measurements



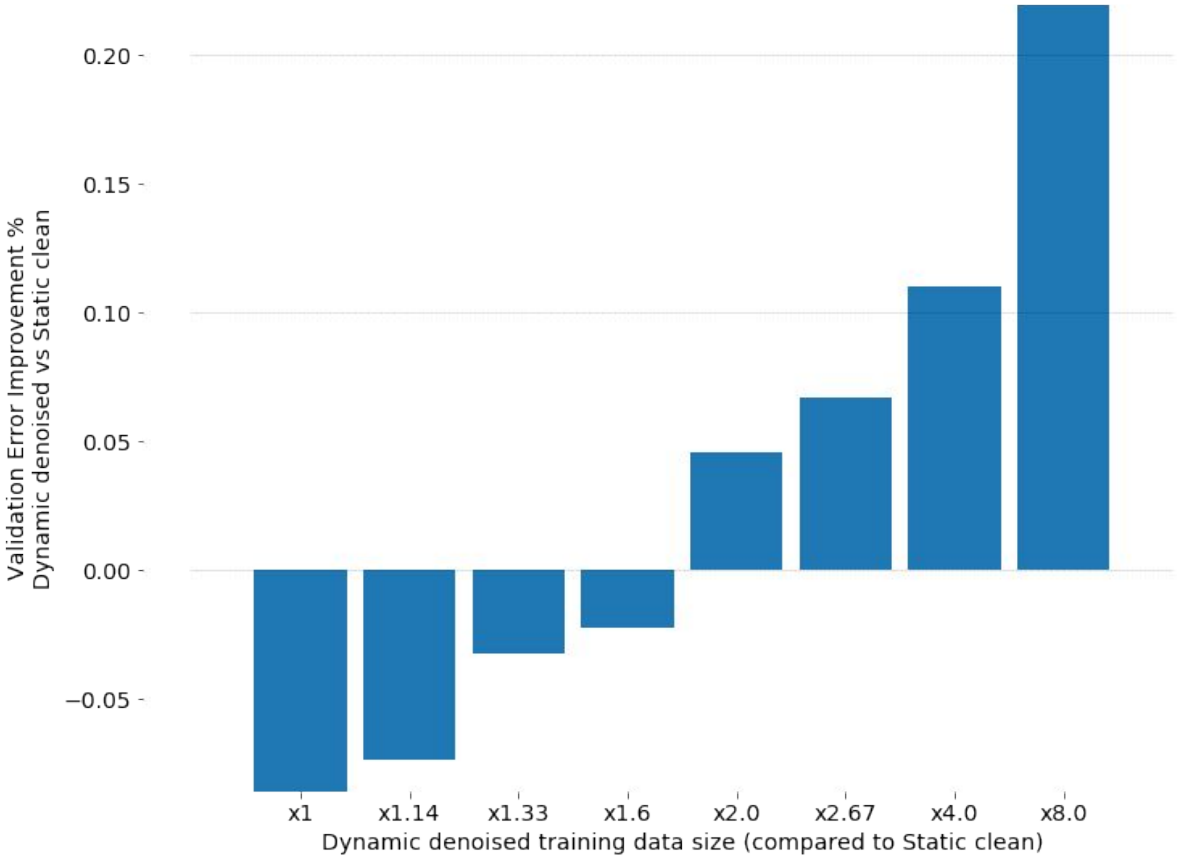
Comparing strategies



Validation Error Comparison

- Fixed training data size
- Fixed number of training cycles/epochs

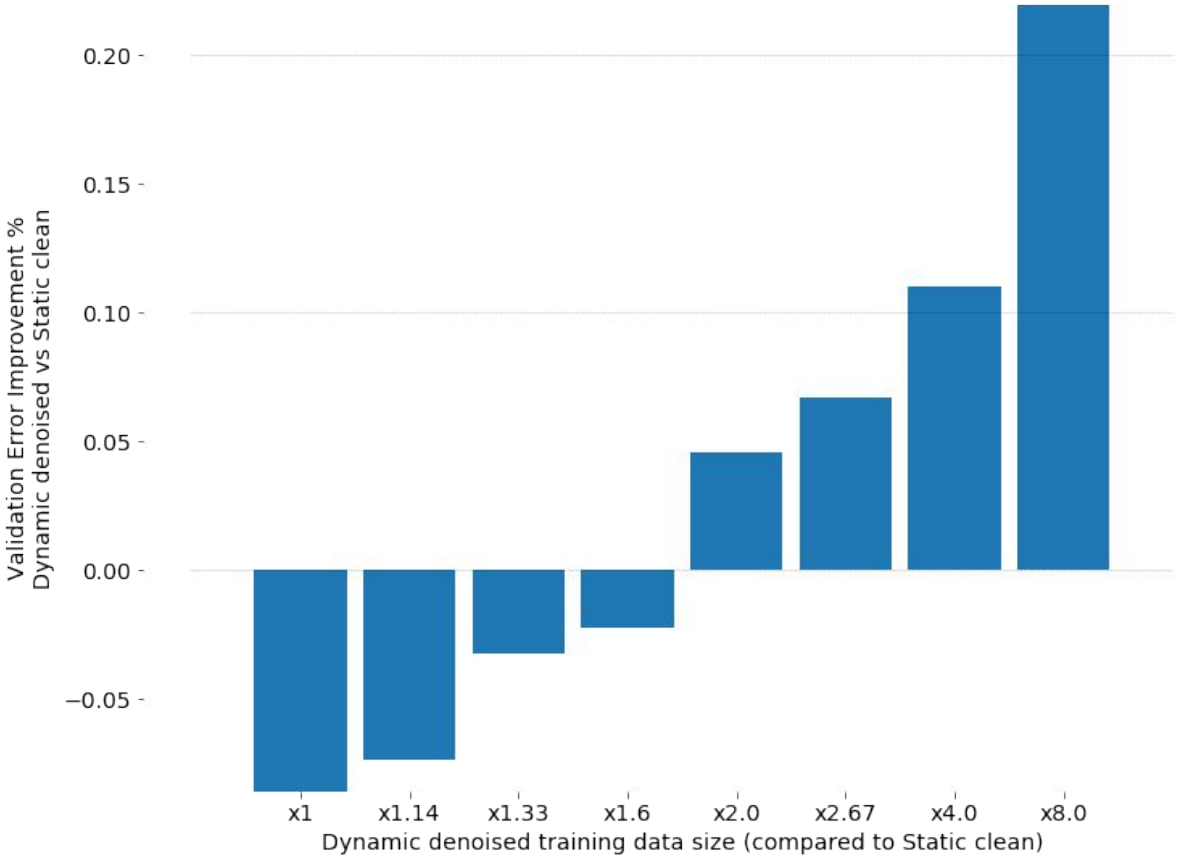
Comparing strategies



- Static model extremely limited in how much training data can be obtained.
- Dynamic denoised can easily be trained on much larger data sets, to give better results.

Meltwater

Comparing strategies

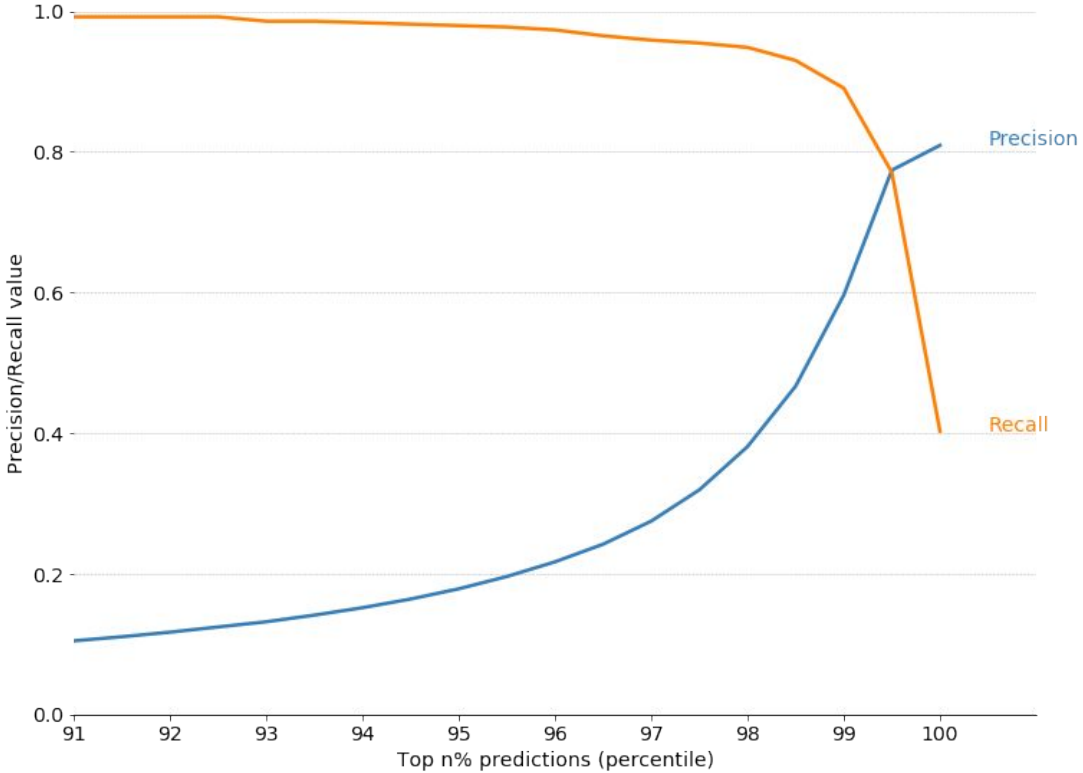


Model Strategies

1. Dynamic noisy model
2. Static clean model
3. Dynamic denoised model

Meltwater

Routing decisions



Can we successfully predict the top 1% (heaviest resource consumption) of queries?

Taking the 99th percentile of predictions, gives:

precision = recall = 0.77

- We detect 77% of the heaviest queries

Future work, Less noise, Better results

- New model features
- Network topology
- Robust regression
  - RANdom Sample Consensus (RANSAC)
  - Ensemble methods

Machine Learning in Information Retrieval

- Improved search result relevance based on user-interactions
- Deep ingressor: generation of ingress/summary for documents
- Deep narration: learn what makes a set of documents different from a background set, and generate a narrative that describes these significant differences
- Provide recommendations on top of search results
- Query augmentation and expansion

Acknowledgements

Special thanks to:

- Meltwater Gothenburg
- Team Horace
- Johan Nilsson Hansen
- Mikael Johansson

Q and A

Meltwater