

Business Intelligence – “Mining”

Everyone thinks of data mining/graphs.

Data mining is a misnomer. It doesn't deal with the actual collection of data.

So this is great ... but what if you have no data?

Open sources of data.

World wide web being one source.

Most web data is in HTML format, and unstructured. It can be gathered with a web spider/crawler.

There's a wikipedia article on Web Mining ...

With a comparison of web mining types.

So there are 4 Types of mining ...

But basically, representations and methods are what we're really concerned about:

Representations:

Bag of words:

“The cat sat on the mat.”

... becomes ...

“The”, “cat”, “sat”, “on”, “the”, “mat”

N-gram terms:

Phonemes, letters, syllables, base pairs.

Phrase/Concept Ontology:

Semantic similarity:

“is-a” type relationship

Semantic relatedness:

Any relationships eg ...

“has-a” type relationships

See:

https://en.wikipedia.org/wiki/Semantic_similarity

English grammar structure

[Common noun](#)

[Proper noun](#)

[Transitive verb](#)

[Intransitive verb](#)

[Adjective](#)

[Adverb](#)

ER structure

Entity type

Entity

Relationship type

Attribute type

Attribute for entity

Attribute for relationship

Source: https://en.wikipedia.org/wiki/Entity%E2%80%93relationship_model

Relational:

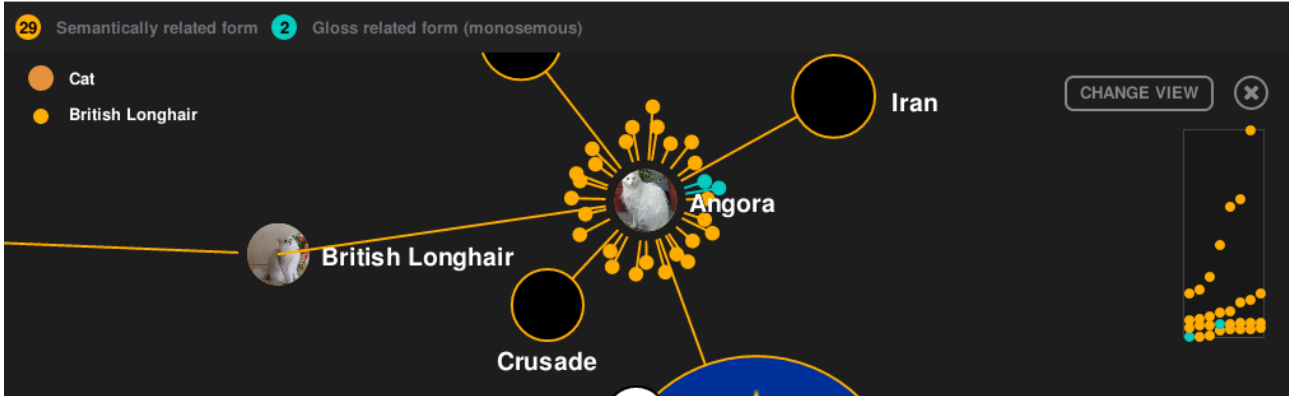
Just a relational database.

Edge Labelled Graph:



BabelNet

Cat ENGLISH TRANSLATE IN... SEARCH



Source: BabelNet.org

Graph:

Just a graph.

Methods:

Machine Learning:

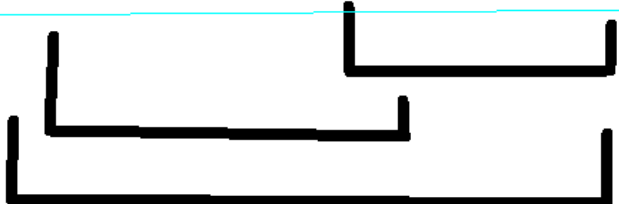
You should know about this already. [To add]

Natural Language Processing:

Rule based:

Nobody follows the rules.

Statistical Based:

"The cat sat on the mat on Jupiter."	
cat mat Jupiter	Nouns
The sat on the on	Relationships
	Statements

General approach:

Find nouns:

Use a dictionary of nouns, for instance:

Wikibase: <https://www.wikidata.org/wiki/Q35120>

For other nouns, use statistical Named-entity recognition:

https://en.wikipedia.org/wiki/Named-entity_recognition

Find relationships:

Find bits in-between nouns, bits preceding the first noun and bits following the last noun.

Find statements[0]:

A loop iterating over relationships, linking two relationships at a time.

Find statements[1]:

A loop iterating over statements[0], linking two statements[0] at a time.

Find statements[2]:

A loop iterating over statements[1], linking two statements[1] at a time.

Find statements[n] where $n > 0$:

A loop iterating over statements[n-1], linking two statements[n-1] at a time

What this can do:

Allows you to type in text like ...

“cats”

Which tells you stuff about cats.

Allows you to type in ...

“cat”, “dog”

Which tells you stuff information that relates cats and dogs, for instance:

“A cat is not a dog.”

“My dog ate the cat.”

etc.

Allows targeted web crawling using a reference document:

Given a document, follow “General approach”, and store results in separate database. Compare the crawled database to the document database. The more the databases match, perhaps the higher likelihood of it being something interesting.

Allows you to select interesting facts and crawl for more.

Self-made search bubble.

Analytics:

Can we record what web-page nouns are recorded on?

Can we record what page relationships are recorded on?

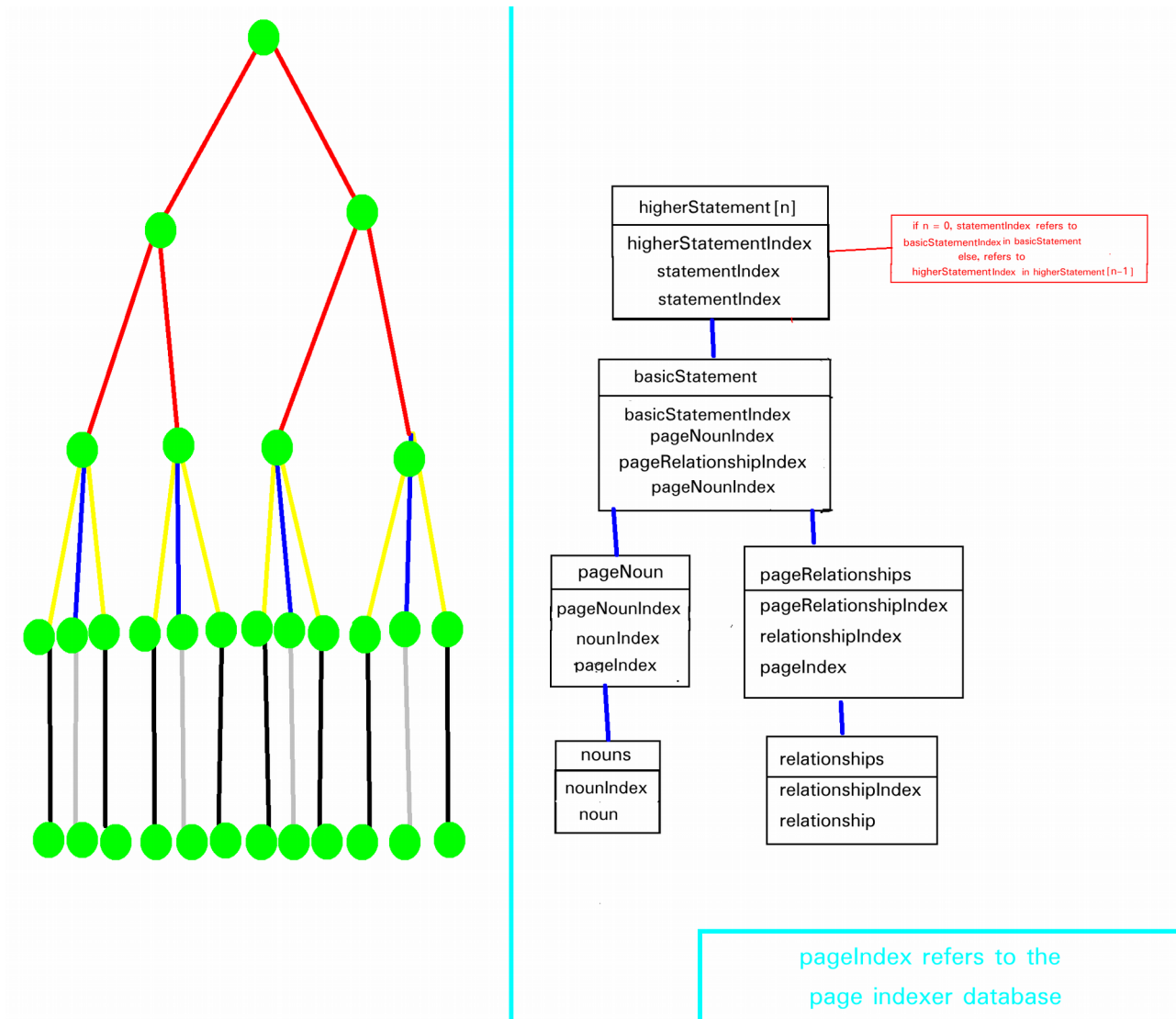
Can we record what page statements are recorded on?

...

Maybe

...

We could lay out the database like this:



We can then count for page, nouns and relationships.

We can also find all the basic statements, and therefore the count.

We can also find all the higher statements and therefore the count.