Emotional Reasoning Chat Bot

Arno Breitfuss, Karen Errou, Juliette Opdenplatz

Overview

Overview

Introduction - Use Case

I feel sad :(make me happy!

Introduction - Use Case

I feel sad :(make me happy!

Watch a happy movie!

Introduction - Domain Analysis

- *Initial thought*:
  Use some movie database.
- Result:
  Movies with ids, director(s), actor(s), genre(s), rating(s), etc.
- Problem:
  How to annotate movies with emotions?

  Genre: Comedy   ==   Happy ?
  Rating: ★★★★☆ ==   Happy ?

Introduction - Domain Analysis

- *Second Approach*:
  Mine sites like **imdb** or **metacritic** in order to obtain movies and <u>reviews</u>.
- Result:
  Same metadata plus reviews which reflect the author's emotion.

  Review: Written text conveys emotions much better than a single value attribute.
                      BUT: Quality of reviews varies (length, subjective/objective, etc.)!

Introduction - Dataset Descriptions

- We mined **imdb** and **metacritic** → 2 named graphs.

metacritic.com	metacritic.com	imdb.com	imdb.com	imdb.com
schema:Movie	onyx: EmotionSet	schema: Person	schema:Movie	onyx: EmotionSet
1	43	6,712	468	8,168

Introduction - Dataset Descriptions

- We mined **imdb** and **metacritic** → 2 named graphs.

metacritic.com	metacritic.com	imdb.com	imdb.com	imdb.com
schema:Movie	onyx: EmotionSet	schema: Person	schema:Movie	onyx: EmotionSet
11,681	110,876	565,063	171,536	1,768,320

Introduction - Initial Vocabulary

- METHONTOLOGY methodology:

1. *Specification*:   Movies have Reviews and Reviews express Emotions.
2. *Conceptualization*: Movies and Reviews via **schema.org**

    Emotions via **Onyx** and **WNAffect** (best suited for our case)
3. *Formalization*:   Movie → Review → Emotion
4. *Implementation*:   Translate mined data to **Turtle**.
5. *Maintenance*:   Added additional metadata (genres and languages) to movies

    using links to wikidata and added some OWL axioms.

Introduction - Knowledge Graph Statistics

- Five weeks ago:

  152,198 triples and 15,449 instances.

- Now:

  28,348,904 triples and 2,627,552 instances.

Overview

1. Introduction
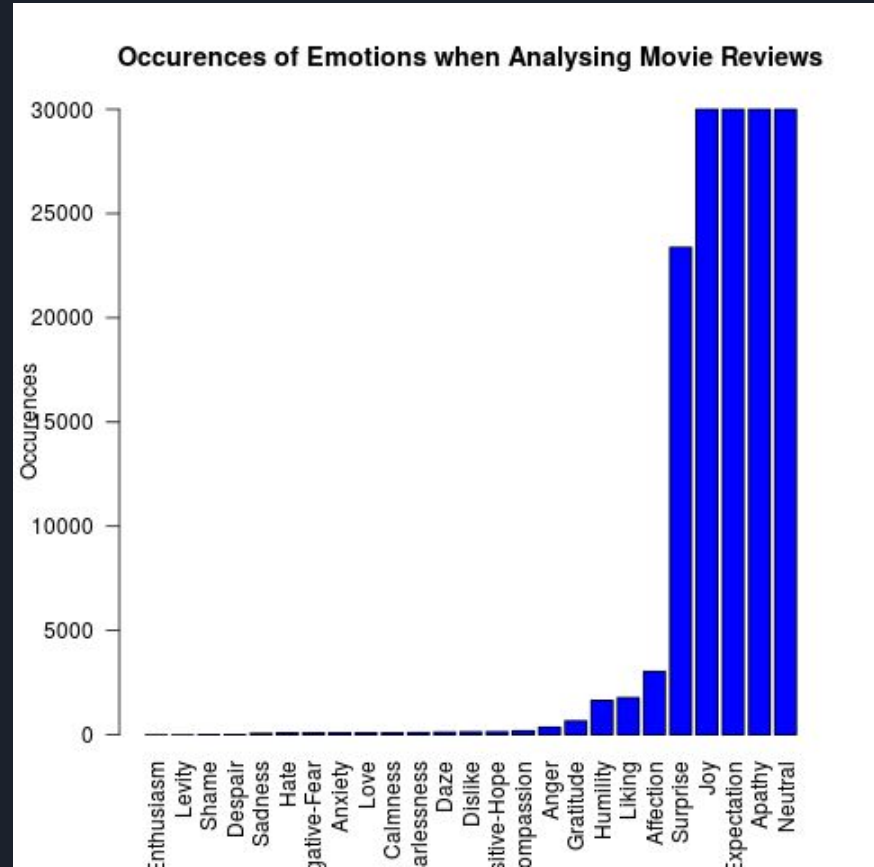2. <u>Enrichment</u>
3. Conclusion and Future Work

Enrichment - Information Extraction

- *Natural Language Annotation*
  - [Ontotext Tagging Service](#)
  - Fully automated process by using the web service
  - Extract:
    - related locations
    - related works
    - related people
    - keyphrases
   - links to the ontotext ontology

Pulp Fiction may be the single best film ever made, and quite appropriately it is by one of the most creative directors of all time, Quentin Tarantino. This movie is amazing from the beginning definition of pulp to the end credits and boasts one of the best casts ever assembled with the likes of Bruce Willis, Samuel L. Jackson, John Travolta, Uma Thurman, Harvey Keitel, Tim Roth and Christopher Walken. The dialog is surprisingly humorous for this type of film, and I think that's what has made it so successful. Wrongfully denied the many Oscars it was nominated for, Pulp Fiction is by far the best film of the 90s and no Tarantino film has surpassed the quality of this movie (although Kill Bill came close). As far as I'm concerned this is the top film of all-time and definitely deserves a watch if you haven't seen it.

Enrichment - Information Extraction

- *Emotion Extraction*
  - <u>Machine Learning</u>
    - Naive Bayes Classifier
    - Thesaurus for training data
  - Obviously not well distributed
    - Training not good enough
    - Method not perfect
  - Solutions
    - Use training data from social media platforms?
    - Add another method

Occurences of Emotions when Analysing Movie Reviews

Enrichment - LOD Links and Alignment
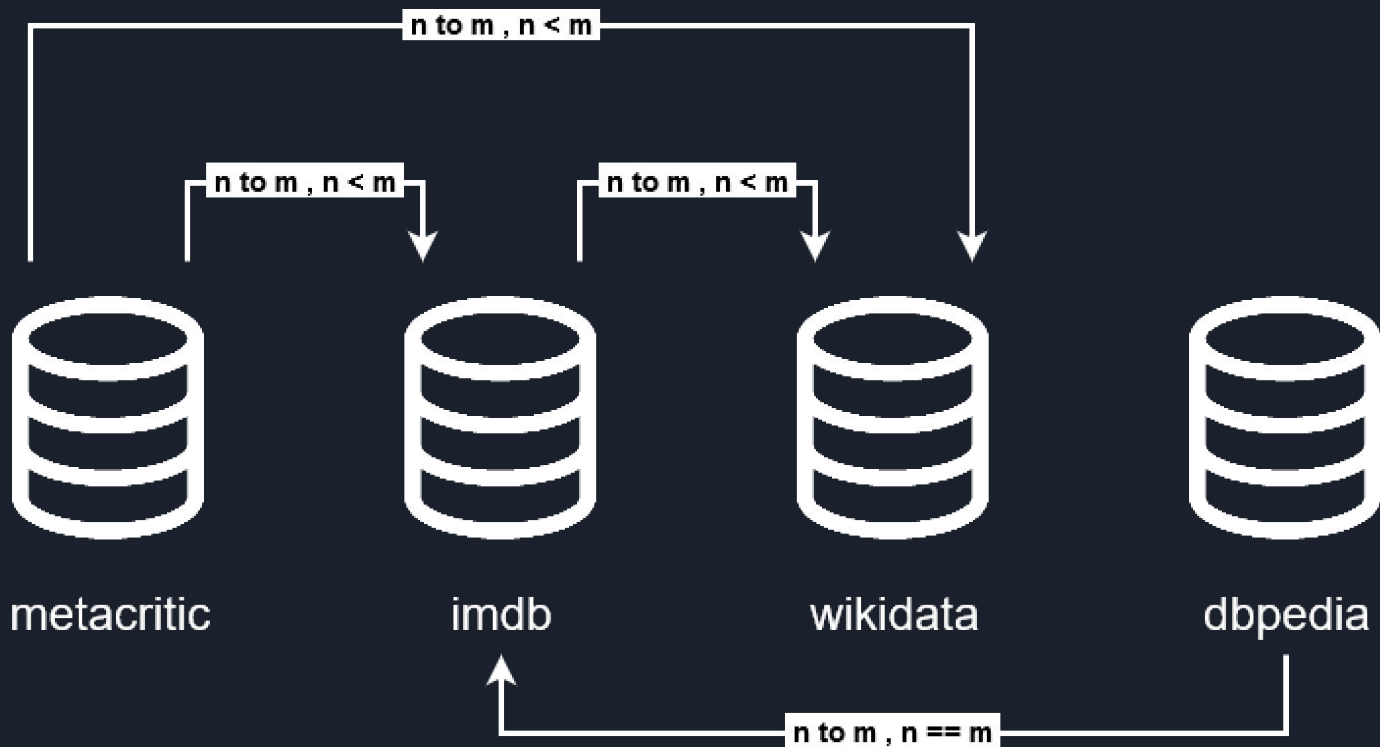
Enrichment - Links For Federated Queries

```sparql
PREFIX schema: <http://schema.org/>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbpedia-owl: <http://dbpedia.org/ontology/>

SELECT DISTINCT ?m ?link ?dbm where {
WHERE {
    SERVICE <http://dbpedia.org/sparql> {
        ?dbm a dbpedia-owl:Film .
        ?dbm owl:sameAs ?link
    }
    ?m a schema:Movie .
    ?m owl:sameAs ?link
}
```

Enrichment - SHACL

SHACL-JS a module for Node.js

- Used by http://shacl.org/playground/
- Fully automated process
  - apply constraints to each .ttl file in specified paths
- https://github.com/TopQuadrant/shacl-js

- Example: At least one genre per movie!

```
[

    a sh:ValidationResult ;
    sh:resultSeverity sh:Warning ;
    sh:sourceConstraintComponent
    sh:MinCountConstraintComponent ;
    sh:sourceShape _:n4343 ;
    sh:focusNode mcb:pulp-fiction ;
    sh:resultPath mcb:hasGenre ;
    sh:resultMessage "A movie should have at least one
    genre!"@en ;
] .
```

Overview

1. Introduction
2. Enrichment
3. <u>Conclusion and Future Work</u>

Conclusion and Future Work

- Summary

1. Data is mined by our own implemented miner
2. Ontology used: Onyx and WNAffect
3. A Crawler is implemented to crawl our data
4. Triple Store used: GraphDB
5. Use of semantic tools: Ontotext, LOD, SHACL

Dataset, Ontologies and Semantic Tools were implemented and used to fulfil the base of our semantic application.

Next Step: Implementing the application

Conclusion and Future Work

- Future Work
1. Let the Chat Bot collect key information about the user emotions.
2. Implementing the Chat Bot using social media applications API (Twitter, Whatsapp, …) or maybe our own.