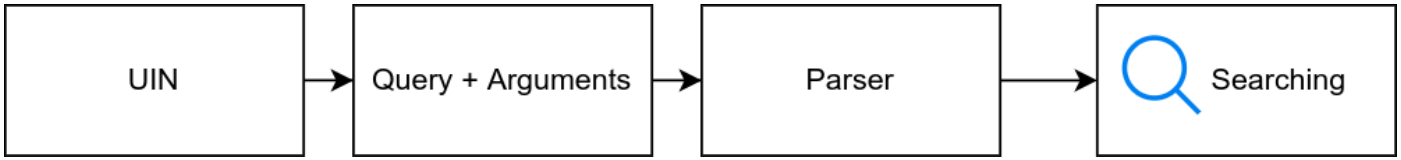
Dati sorgenti

- **Text Classification Model per Sentiment Analysis è accessibile via**: huggingface.co/distilbert-base-uncased-finetuned-sst-2-english
- **Binary classification (Positive/Negative)**

- **Opinion-lexicon-English dictionary per Lexicon-based Sentiment Analysis è accessibile via cs.uic.edu/~liub/FBS/sentiment-analysis.html**
- **Dictionary is ordered in alphabetical order**
- **2 Files: Positive-Words.txt & Negative-Words.txt**

Linguaggio di interrogazione



Tipi di argumenti supportati:

--Lsentiment positive/negative → Model-based Sentiment Analysis
--Msentiment positive/negative → Lexicon-based Sentiment Analysis
--mf True/False → Multi-field Search

Types of queries supported:

- **Boolean: Keywords + (AND, OR, NOT)**
- **Phrase: Ordered list of contiguous words**
- **Proximity: Max allowed distance between words in the query**
- **Wildcard queries**

Support of 'Did you mean?'

```
[>>>] Searching for: adventare --Msentiment positive
[>>>] Querying: workName:adventare
Empty result!!
   [?] Did you mean adventure ? adventures ?
```

Query example: **"Witch Hunter" or "is part" --Msentiment negative --mf True**

Schema

```
schema = Schema(
    workName=TEXT(analyzer=analysis.StandardAnalyzer(), stored=True),
    workId=ID(stored=True),
    review=TEXT(stored=True, analyzer=analysis.StemmingAnalyzer()),
    sentimentLabel=ID(stored=True, sortable=True),
    sentimentScore=NUMERIC(stored=True, sortable=True, signed=False),
    sentimentLabelLexiconAnalysis=ID(stored=True, sortable=True),
    sentimentScoreLexiconAnalysis=NUMERIC(stored=True, sortable=True, signed=False),
)
```

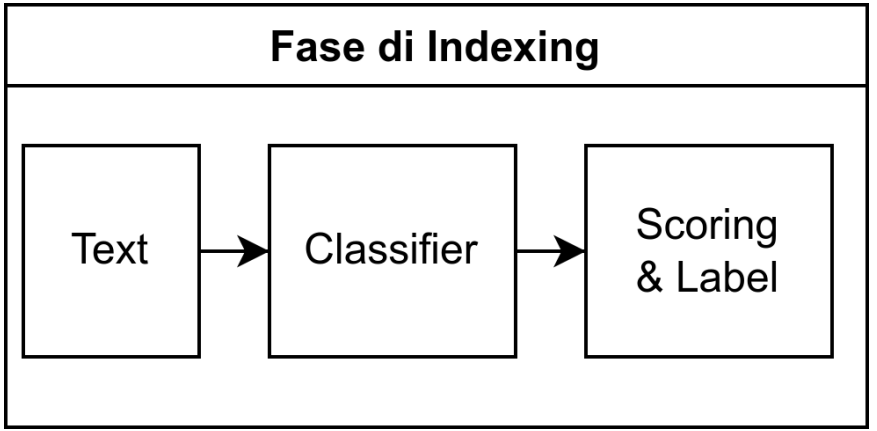
workName - Nome del cartone animato
workID - ID del cartone animato su myanimelist.net/anime/'ID' (to be stored/indexed) + StandardAnalyzer
review - Comment content (to be stored/indexed) + StemmingAnalyzer

SentimentLabel - Model output sentiment type: Negative/Positive
SentimentScore - Model output sentiment score (>=0)

SentimentLabelLexiconAnalysis - Lexicon-Based output sentiment type
SentimentScoreLexiconAnalysis - Lexicon-Based output sentiment score

Tecniche di sentiment analysis usate

1) Model-based



Vantaggi:

- **Buona accuratezza rispetto a Lexicon-based sentiment**
- **Veloce**

Svantaggi:

- **Sensibile al rumore (noise) - carratteri non ASCI**
- **Limite alla lunghezza del testo →** **Soluzione:**

```
classifier = pipeline("sentiment-analysis",
model='distilbert-base-uncased-finetuned-sst-2-english')
result = classifier(row['review'], truncation=True)
```

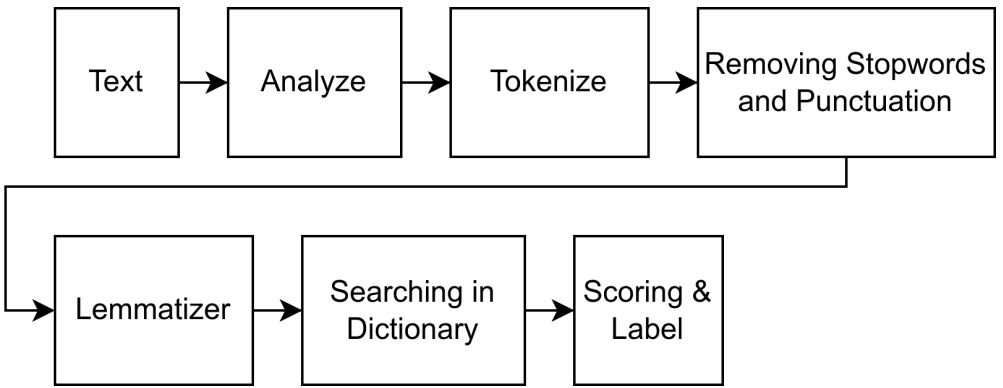
.. or chunking

Traceback (most recent call last):
Token indices sequence length is longer than the specified maximum sequence length for this model (815 > 512). Running this sequence through the model will result in indexing errors

Tecniche di sentiment analysis usate

2) Lexicon-based sentiment analysis that calculates the sentiment score using words present in a text.

- **Binary Search can (must) be exploited due to ordered structure of dictionary**



$$Scoring = \frac{(\text{number of positive words})}{(\text{number of negative words} + 1)}$$

$Scoring < 0,98 \longrightarrow \text{NEGATIVE}$

$Scoring \geq 0,98 \longrightarrow POSITIVE$

Can be processed in Parallel

Complexity? $O(k) * O(\log(n)) = O(k \log(n))$

Input size: k terms, n numero di terms in dizionario

```
for w in cleaned_sentence:
    if self.BinarySearch(w, self._NEGATIVE_WORDS):
        SentimentN += 1
    if self.BinarySearch(w, self._POSITIVE_WORDS):
        SentimentP += 1
```

Alternative scoring:

$$\frac{(\text{number of positive words}) - (\text{number of negative words})}{(\text{total number of words})}$$

$$(\text{number of positive words}) - (\text{number of negative words})$$

Tecniche di sentiment analysis usate

2) Lexicon-based sentiment analysis that calculates the sentiment score using words present in a text.

```
I don't like this TV program
```

Score: 1.0 Label: POSITIVE

Svantaggi:

- **Isolated word processing (non considera il contesto circostante)**
- **Mantenere il dizionario aggiornato con parole nuove**

- **Possiamo migliorare la tecnica di Lexicon-based, definendo un insieme di regole per la fase di preprocessing:**
  1. **Negation**
  2. **Sentence & Clause analysis**
  3. **...**

Scoring

- **The whoosh.scoring module contains implementations of various models for ranking:**

TF-IDF

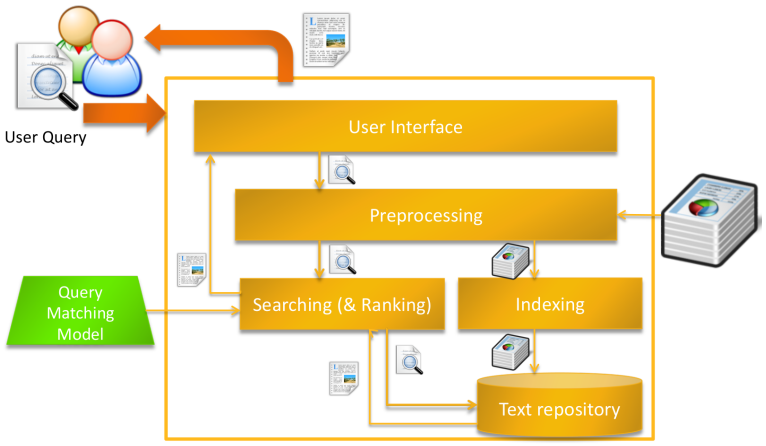
- **Option: --Msentiment**

BM25

- **Option: --Lsentiment**

Architettura IR

Store first-query later technology



Query Processing:

Query Flow:
User types query in UI → Preprocessing → Searching (& Ranking) → QMM outputs relevant results for user query → Output results in UI

Data Flow:
Preprocessing → Indexing → Text repository

How much time do we need to index 10000 records?

It depends on many factors such as:

- **Preprocessing for indexing**
- **Complexity of sentiment analysis algorithm**
- **Type of secondary memory used**
- **Complexity of Scoring function**
- **...**

- **Empirical data (my PC):**

Elapsed time: 6540.69 sec ~ 1 Hour and 49 Minutes

→ 0,654 sec per record

Ranking

- **Custom ranking (eredità dalla classe WeightingModel)**

Perché fare custom Ranking?

- **Risultati con sentimento cercato saranno premiati (saranno più rilevanti in ranking)**
- **Definisce un ordinamento totale basato sia sul contenuto testuale sia sul sentimento espresso**
- **Diverse configurazioni di modelli IR + Diversi tecniche di sentiment analysis = varietà**

Benchmark

- **10 User Information Need tradotti in linguaggio di interrogazione del IR system**

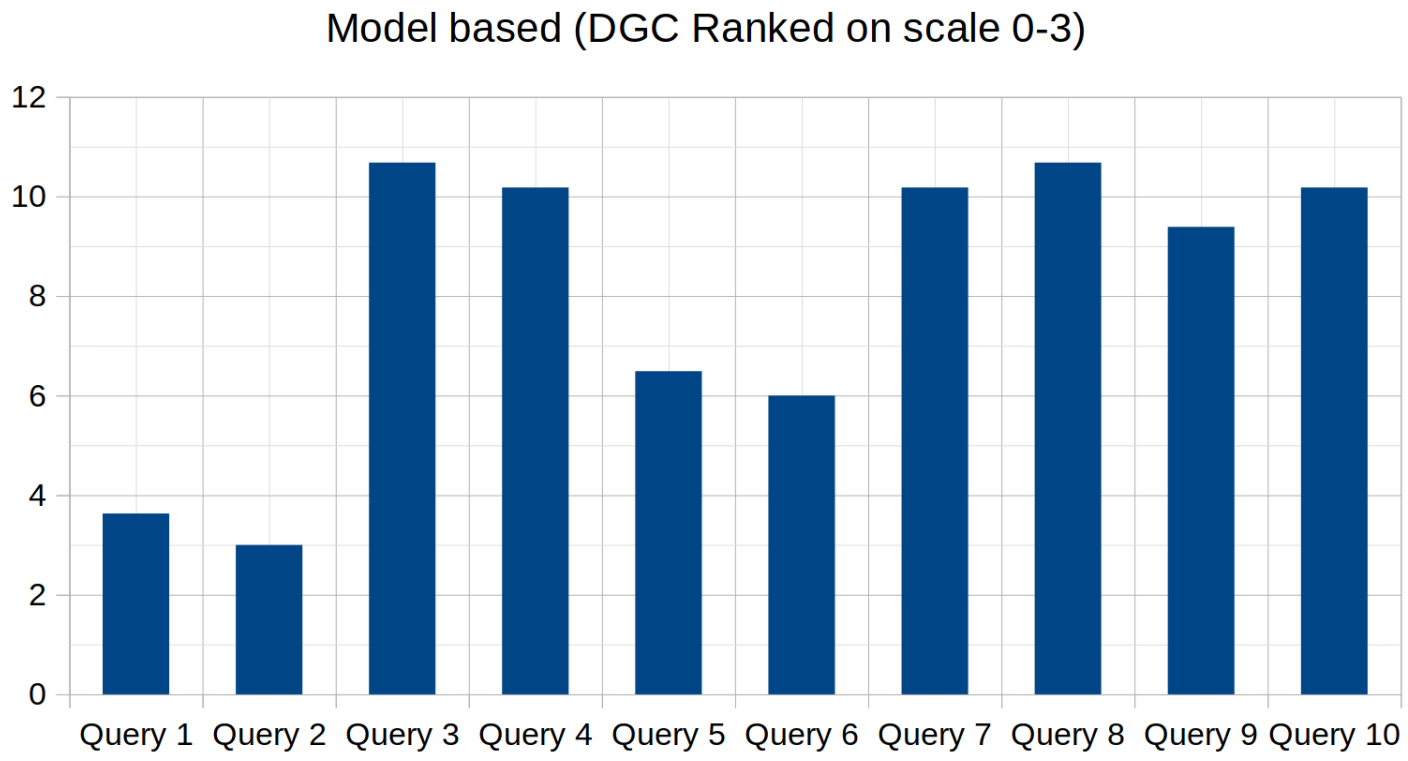
Misura di performance utilizzata per valuare efficacia di IR system?

- **DGC (Discounted Cumulatative Gain):**
  - Rilevanza sulla scala da 0-3
  - Profondità fino a 5 documenti

$$DCG_p = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{\log_2 i}$$

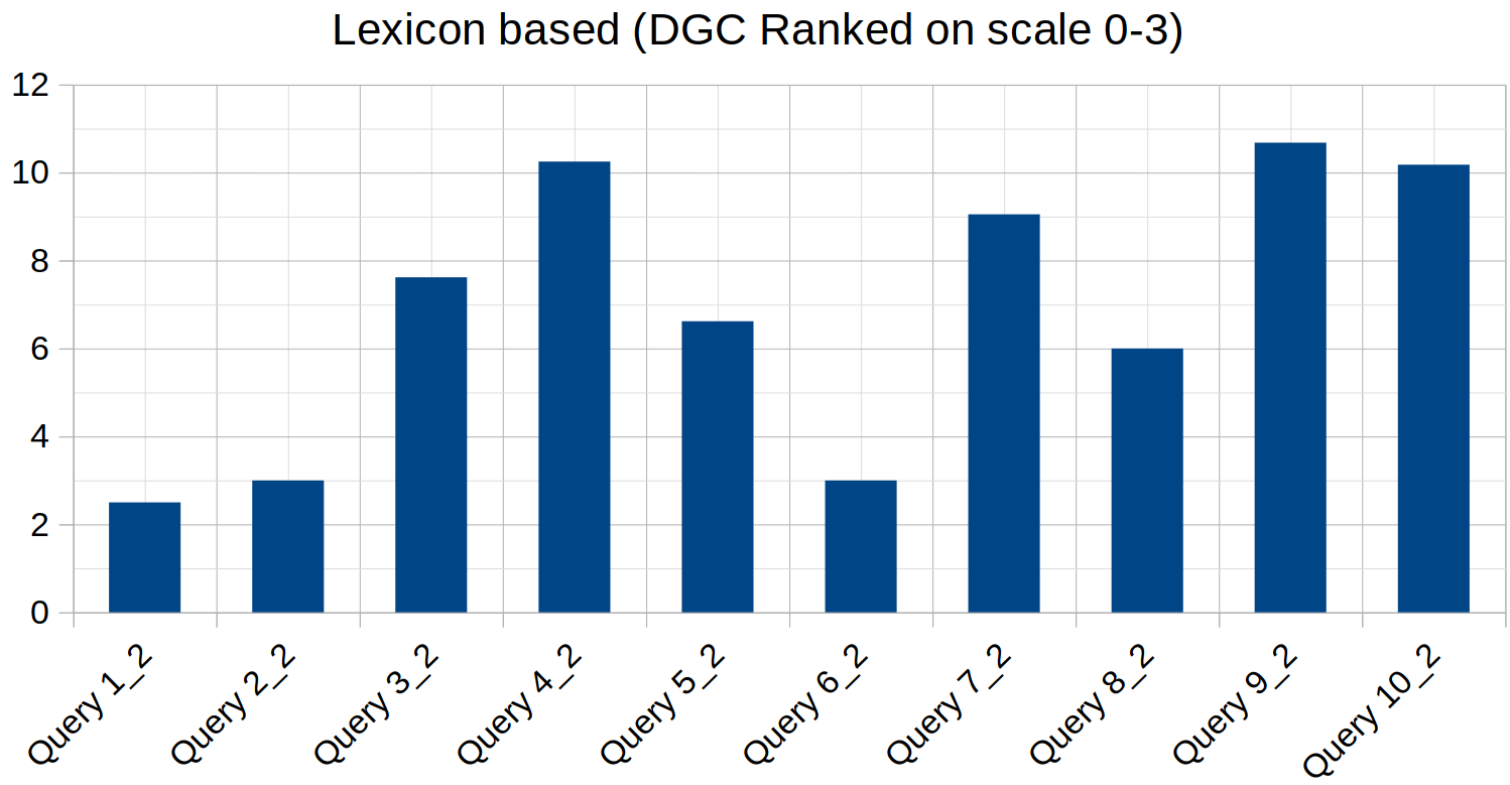
Benchmark

- **Rilevanza sulla scala da 0-3**
- **Ranking = TF-IDF + ModelScore**
- **Option: --Msentiment**

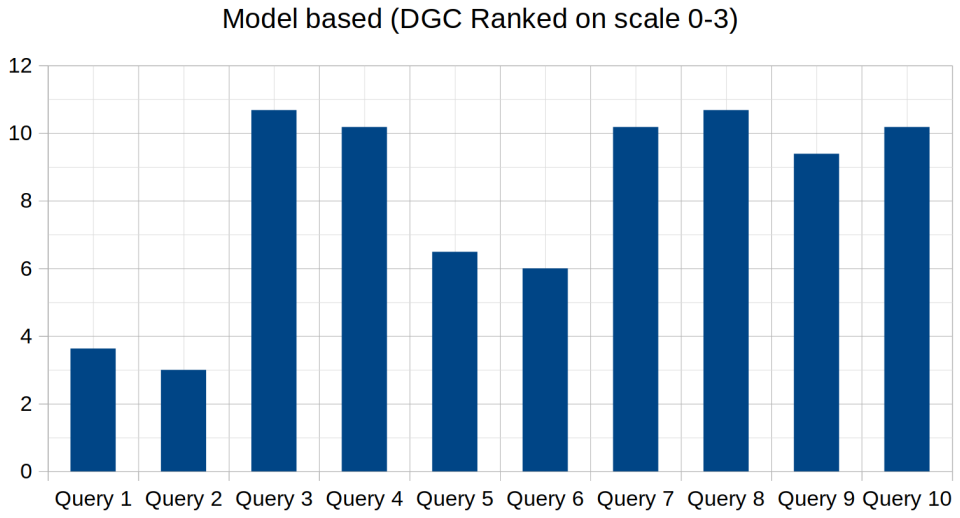


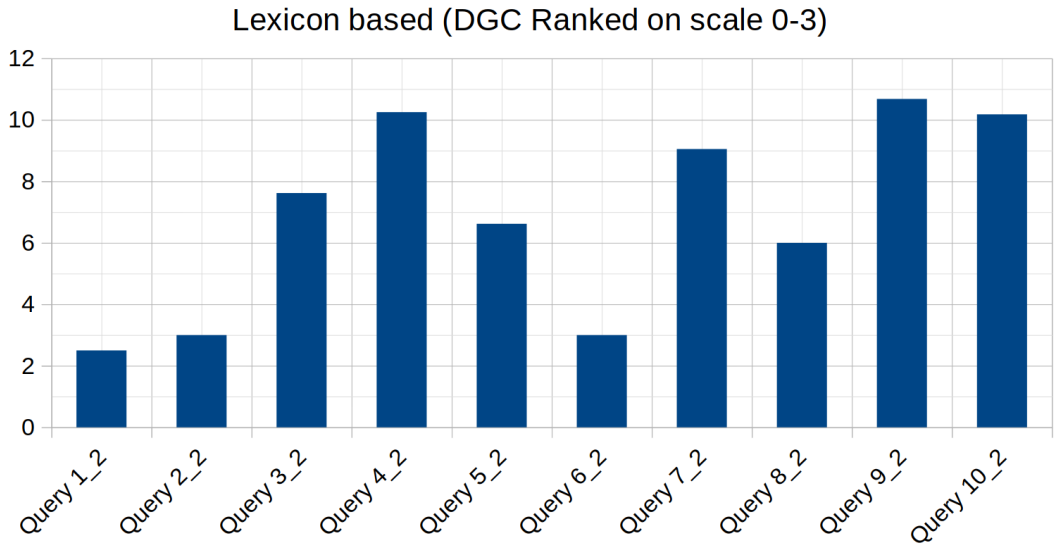
Benchmark

- **Rilevanza sulla scala da 0-3**
- **Ranking = BM25 + LexiconScore**
- **Option: --Lsentiment**



Benchmark





- **Model-based restituisce più risultati rilevanti per gli UINs delle query 6, 7 e 8**
- **Per le query 6, 7 e 8 Model-based vince con maggiore margine in quanto è più preciso a stabilire sentimento rispetto a Lexicon-based** **Dizionario non è abbastanza ricco?**
- **Per la query 9 Lexicon-based ha un piccolo margine rispetto a Model-based**