Information Retrival Project

– Neural Networks in Information Retrival –

Flavia De Santis

A.A. 2023/2024

Theory:

- Definition of Text Retrieval Tasks
- Text Representation with NN
- Deep NN for IR: Architectures and Techniques

Project:

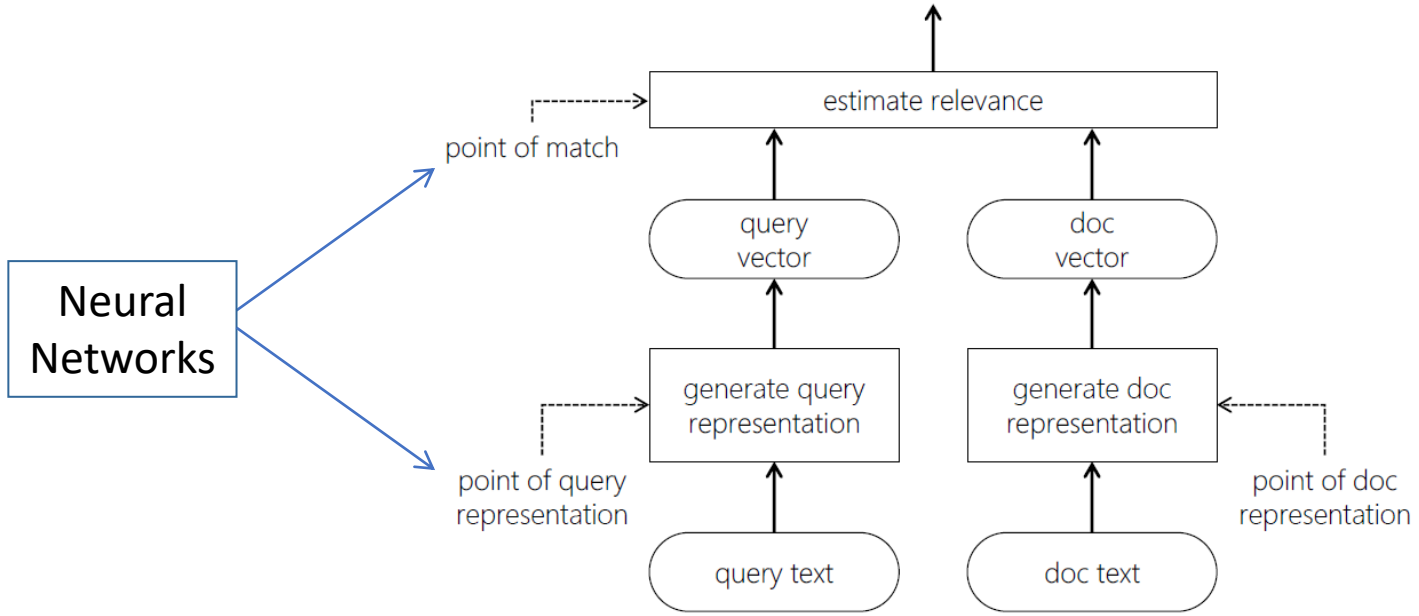
- Dataset (from Kaggle)
- Pre-processing and implementing a CNN-based Model
- Implementation Details
- IR task
  - Query
  - Reviews ranking
- Conclusions

Definition of Text Retrival Task

- **Ad-hoc retrieval**
- **Question-answering**

Desiderata

- Semantic understanding
- Robustness to …
- Sensitivity to context
- Efficiency



Source: Mitra and Craswell (2018)

Document ranking typically involves a query and a document representation steps, followed by a matching stage. Neural models can be useful either for generating good representations or in estimating relevance, or both.

Definition of Text Retrival Task (2)

Metrics

$$Precision_q = \frac{\sum_{\langle i,d\rangle \in R_q} rel_q(d)}{|R_q|}$$

$$Recall_q = \frac{\sum_{\langle i,d\rangle \in R_q} rel_q(d)}{\sum_{d \in D} rel_q(d)}$$

$$RR_q = \max_{\langle i,d\rangle \in R_q} \frac{rel_q(d)}{i}$$

$$AveP_q = \frac{\sum_{\langle i,d\rangle \in R_q} Precision_{q,i} \times rel_q(d)}{\sum_{d \in D} rel_q(d)}$$

$$\mathrm{DCG}_q = \sum_{\langle i,d\rangle \in R_q} \frac{2^{rel_q(d)} - 1}{\log_2(i+1)}$$

Traditional IR models

- BM25
- LM

} # of query term occurrences in the document

- Dependence model → phrasal matches

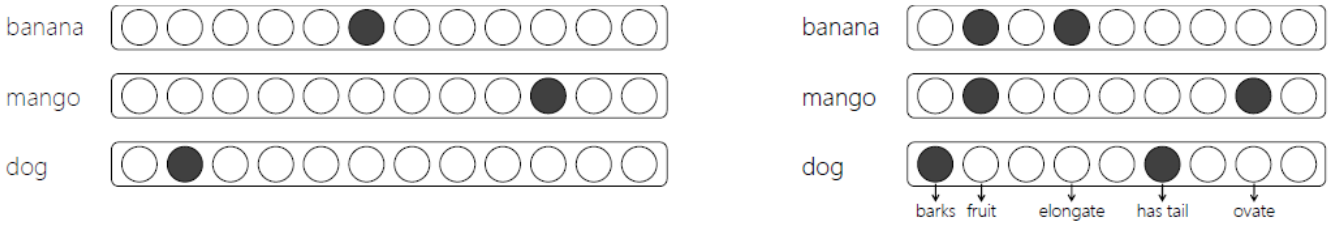
- Translation models
- PRF

} vocabulary mismatch bw query and document

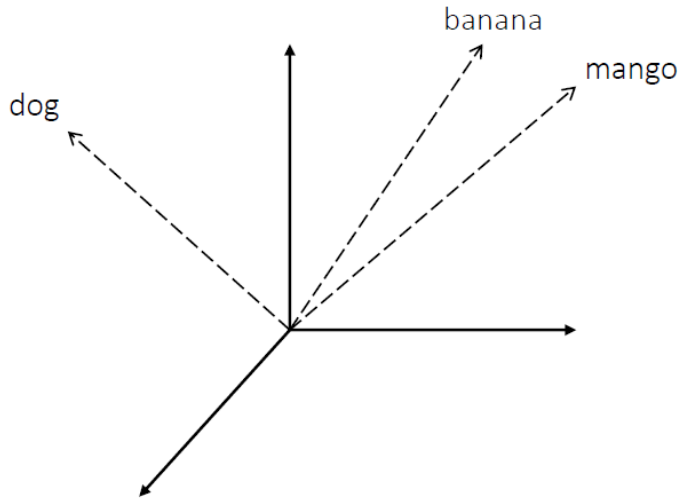
COSINE SIMILARITY

$$sim(\vec{v}_i, \vec{v}_j) = cos(\vec{v}_i, \vec{v}_j) = \frac{\vec{v}_i^{\mathsf{T}} \vec{v}_j}{\|\vec{v}_i\| \|\vec{v}_j\|}$$

Text Representation with Neural Networks



(a) Local representation (b) Distributed representation



Each **term** can be represented within a **neural network** by the activation of a **single** neuron (**local representation**) or by the pattern of activations of **several** neurons (**distributed representation**)

Source: Mitra and Craswell (2018)

Text Representation with Neural Networks (2)

Observed feature spaces

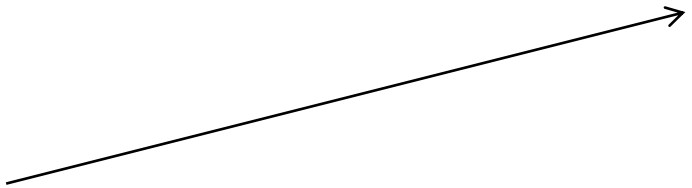
SPARSE

vs

DENSE

Latent feature spaces

- Latent Semantic Analysis (LSA)
- Probabilistic LSA (PLSA)
- Word2vec
- GloVe
- Paragraph2vec

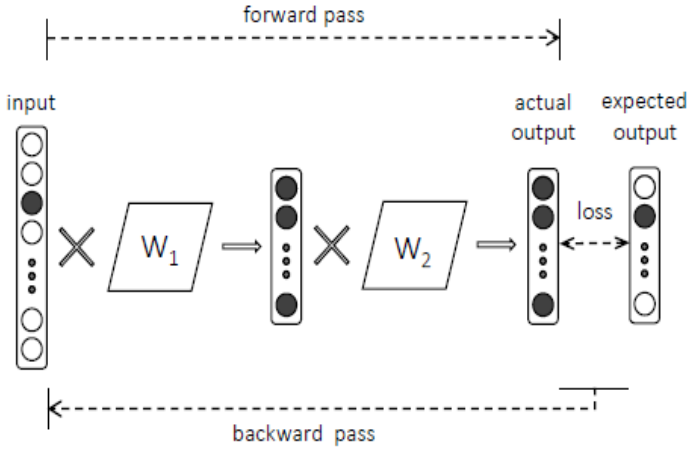


An **embedding** is a representation of items in a new space s.t. the original representation's properties of the items are preserved

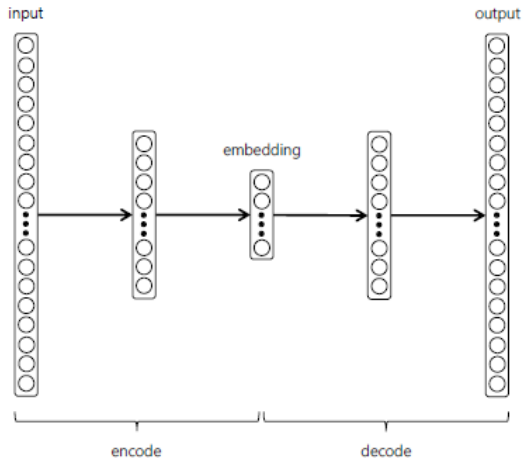
Neural IR models can influence

- the **query** representation
- the **document** representation
- the **relevance** estimation

Deep Neural Networks for IR: Architectures and Techniques

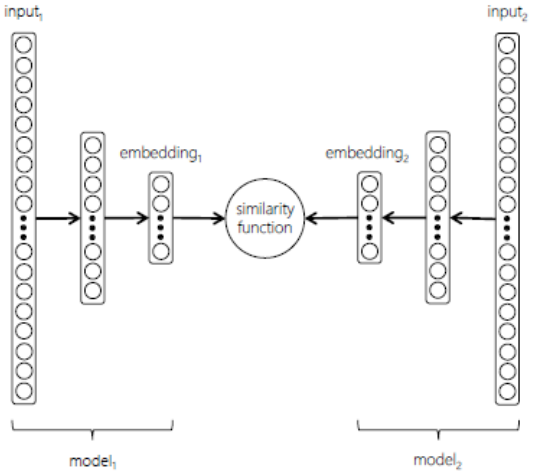


Feed-forward neural network with a single hidden layer.



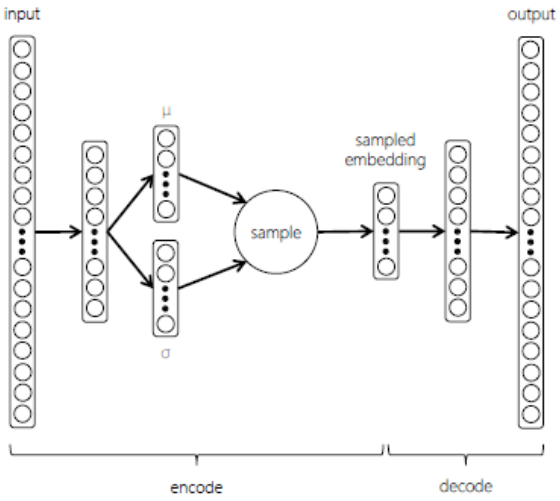
AUTOENCODER

compressed representations of inputs



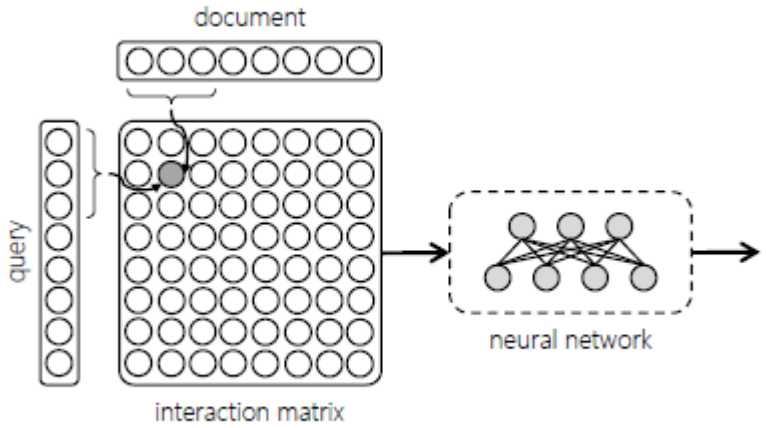
SIAMESE NETWORK

VARIATIONAL AUTOENCODER

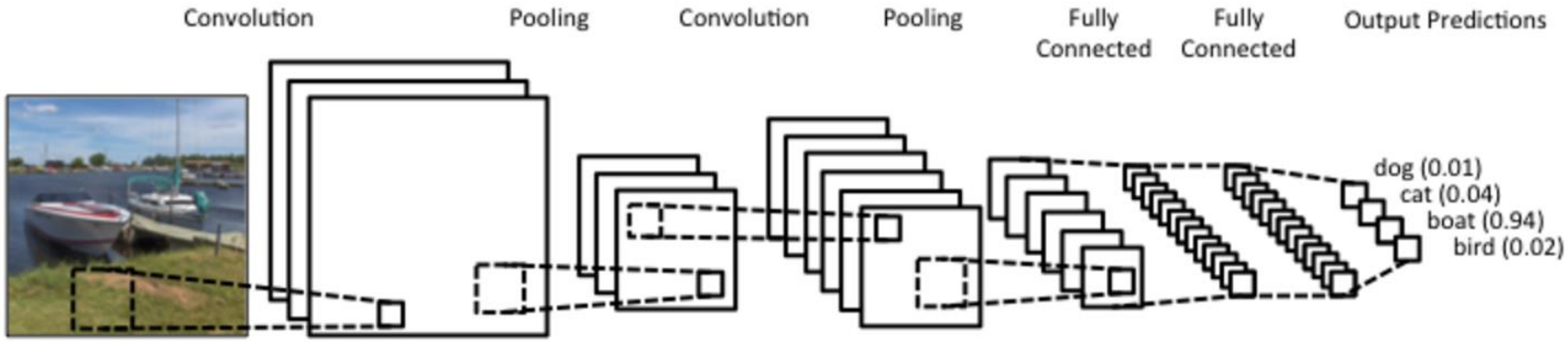


Source: Mitra and Craswell (2018)

Deep Neural Networks for IR: Architectures and Techniques (2)



Interaction matrix generated by comparing windows of text from the query and the document. A DNN operates over the interaction matrix to find patterns of matches that suggest relevance of the document to the query.



Dataset (from Kaggle)

	user_id	user_sex	user_age	user_country	rating	comment	favourite_character	date
1	0	female	50	Germany	2.5	ndtrack was forgettable."	Severus Snape	2004-12-27
2	1	female	23	Spain	4	adds an intriguing layer."	Severus Snape	2003-11-22
3	2	male	32	France	3	aracters were charming."	Ron Weasley	2005-09-16
4	3	female	24	Turkey	4.5	atures is heartwarming."	Rubeus Hagrid	2002-09-17
5	4	female	40	Spain	5	ourage is awe-inspiring."	Neville Longbottom	2004-10-17
6	5	female	16	Finland	5	creatures is endearing."	Rubeus Hagrid	2002-12-02
7	6	male	26	Germany	4	adds depth to the story."	Severus Snape	2005-11-28
8	7	female	28	Sweden	2.5	cessary and distracting."	Albus Dumbledore	2005-02-10
9	8	female	52	Switzerland	4	adds a delightful touch."	Ron Weasley	2002-02-02
10	9	female	46	Turkey	4.5	termination is inspiring."	Hermione Granger	2005-12-02

...

comment
"The transitions between scenes were awkward, and the soundtrack was forgettable."
"Severus Snape's role adds an intriguing layer."
"The pacing was a bit slow, but the characters were charming."
"Hagrid's love for magical creatures is heartwarming."
"Neville Longbottom's courage is awe-inspiring."
"Rubeus Hagrid's love for magical creatures is endearing."
"Severus Snape's complexity adds depth to the story."
"Albus Dumbledore's presence feels unnecessary and distracting."
"Ron Weasley's humor adds a delightful touch."
"Hermione Granger's determination is inspiring."

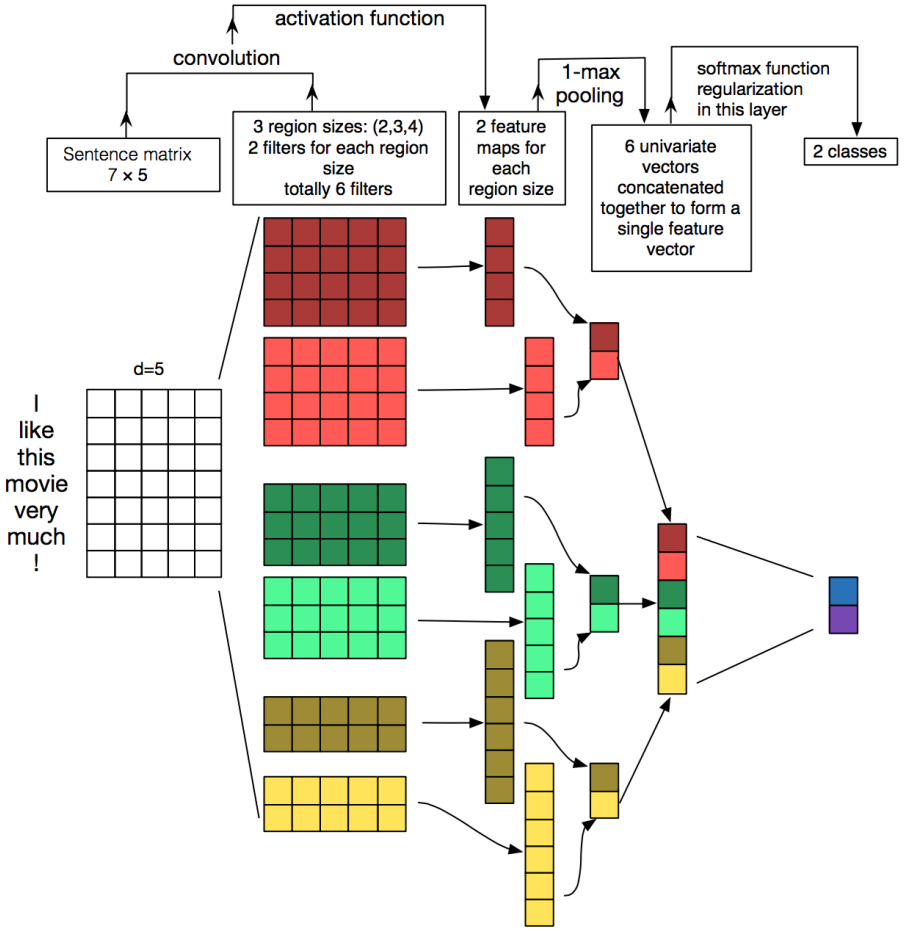
...

Dataset

	user_id	user_sex	user_age	user_country	rating	comment	favourite_character	date
1	0	female	50	Germany	2.5	ndtrack was forgettable."	Severus Snape	2004-12-27
2	1	female	23	Spain	4	adds an intriguing layer."	Severus Snape	2003-11-22
3	2	male	32	France	3	aracters were charming."	Ron Weasley	2005-09-16
4	3	female	24	Turkey	4.5	atures is heartwarming."	Rubeus Hagrid	2002-09-17
5	4	female	40	Spain	5	ourage is awe-inspiring."	Neville Longbottom	2004-10-17
6	5	female	16	Finland	5	l creatures is endearing."	Rubeus Hagrid	2002-12-02
7	6	male	26	Germany	4	y adds depth to the story."	Severus Snape	2005-11-28
8	7	female	28	Sweden	2.5	cessary and distracting."	Albus Dumbledore	2005-02-10
9	8	female	52	Switzerland	4	adds a delightful touch."	Ron Weasley	2002-02-02
10	9	female	46	Turkey	4.5	termination is inspiring."	Hermione Granger	2005-12-02

...

Pre-processing and implementing a CNN-based Model



Source: Zhang, Y., & Wallace, B. (2015)

Pre-processing

```
# Tokenize the text and convert it into sequences
tokenizer = Tokenizer()
tokenizer.fit_on_texts(harry_potter_reviews['comment'])
X = tokenizer.texts_to_sequences(harry_potter_reviews['comment'])
# Pad sequences to ensure uniform length
X = pad_sequences(X)
# Encode target labels
label_encoder = LabelEncoder()
y = label_encoder.fit_transform(harry_potter_reviews['favourite_character'])
```

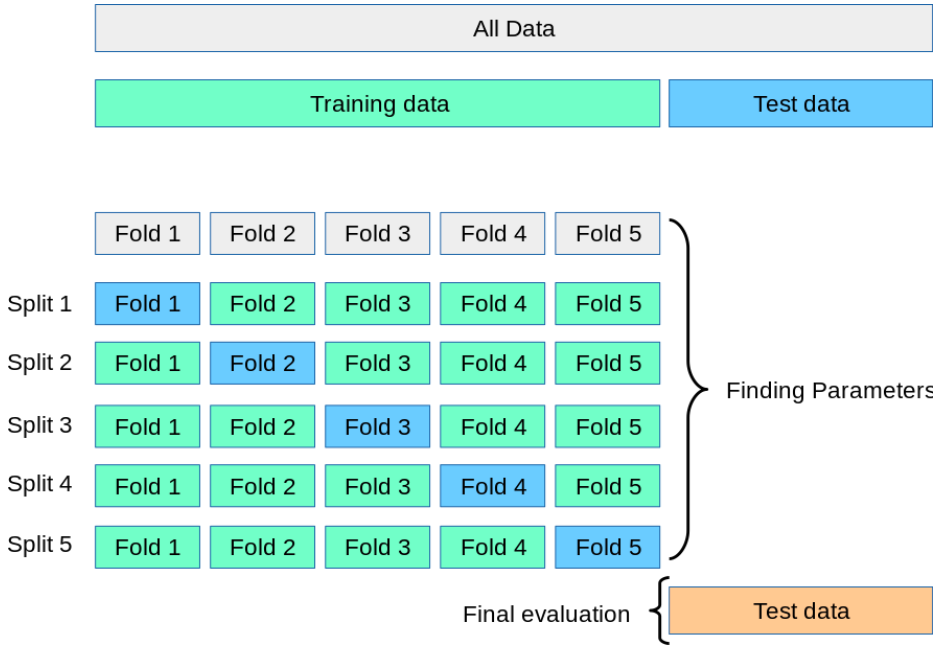
Implementing the model

```
# Model creation function
def create_model():
    # Define CNN architecture
    model = Sequential()
    model.add(Embedding(input_dim=len(tokenizer.word_index)+1, output_dim=200,␣
 ↪input_length=X.shape[1]))
    model.add(Conv1D(128, 5, activation='relu'))
    model.add(GlobalMaxPooling1D())
    model.add(Dense(16, activation='relu'))
    model.add(Dense(len(label_encoder.classes_), activation='softmax'))
    model.add(Dropout(0.3))  # Added Dropout layer

    model.compile(loss='sparse_categorical_crossentropy', optimizer='adam',␣
 ↪metrics=['accuracy'])
    return model
```

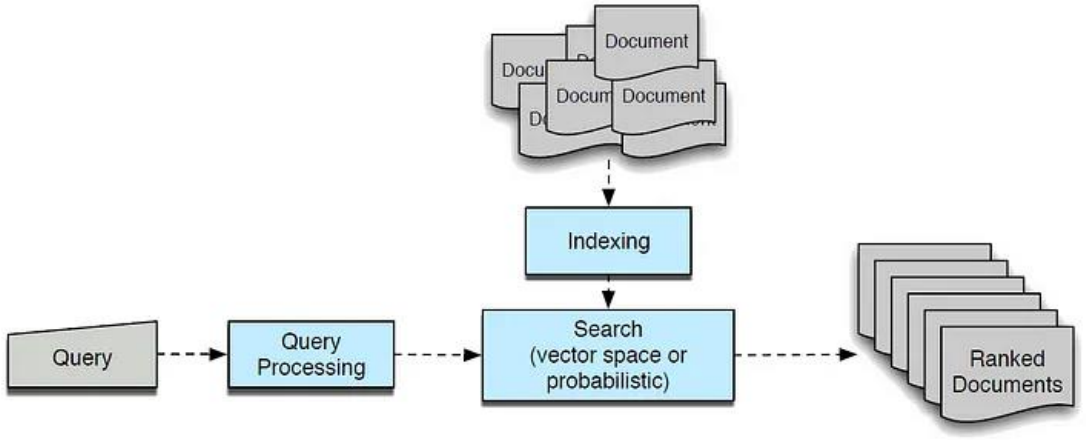
Implementation Details

```
# K-fold cross-validation
k_fold = 5
kf = KFold(n_splits=k_fold)
# Empty list for the cv scores
cv_scores = []
for train_index, test_index in kf.split(X):
    # Split data into training and testing sets
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]
    # Create and train the model
    model = create_model()
    model.fit(X_train, y_train, epochs=8, batch_size=32, verbose=0)  #␣
↪verbose=0 to suppress training output
    # Evaluate the model
    score = model.evaluate(X_test, y_test, verbose=0)
    cv_scores.append(score[1])
```



Source: Sklearn

IR task: query



Source: medium.com

```
# Define the query
query = "Ron Weasley is funny!"

# Tokenize and pad the query sequence to match the input shape of the model
query_sequence = tokenizer.texts_to_sequences([query])
query_padded = pad_sequences(query_sequence, maxlen=X.shape[1])

# Obtain the embedding for the query using the trained model
query_embedding = model.predict(query_padded)

# Compute cosine similarity between the query embedding and embeddings of all␣
↪reviews in the dataset
similarities = cosine_similarity(query_embedding, model.predict(X))

# Find the index of the most similar review
most_similar_index = similarities.flatten().argsort()[-1]

# Retrieve the most similar review from the dataset
most_similar_review = harry_potter_reviews['comment'].iloc[most_similar_index]

print("Most similar review:", most_similar_review)
```

```
1/1 [==============================] - 0s 47ms/step
16/16 [==============================] - 0s 7ms/step
Most similar review: "An epic adventure with humor and heart."
```

IR task: reviews ranking

```python
# Set maximum length for padded reviews
max_length = 100

def get_reviews_ranking_custom(query, harry_potter_reviews, model, tokenizer):
    '''
    Function to rank reviews based on a query using a CNN model
    '''
    # Tokenize and filter stopwords for query
    tokenized_query = word_tokenize(query)
    stop_words = set(stopwords.words('english'))
    filtered_query = [word for word in tokenized_query if word.lower() not in␣
 ↪stop_words]
    query_text = ' '.join(filtered_query)
    # Get all reviews from the dataset and append the query

    # Encode and pad all reviews

    # Obtain embeddings for all reviews

    # Separate query embedding from the rest

    # Compute cosine similarity between query embedding and all review␣
 ↪embeddings

    # Rank reviews based on similarity scores

    return reviews_ranking
```

```python
# Your query
your_query = "hermione's smartness is well-represented."

# Get ranking of reviews for your query
ranking = get_reviews_ranking_custom(your_query, harry_potter_reviews, model,␣
 ↪tokenizer)

# Print the ranking of reviews
print("Ranking of reviews for your query:", ranking['comment'])

# Save ranking to CSV file
output_file = "ranking_results.csv"
ranking.to_csv(output_file, index=False)
```

```
16/16 [------------------------------] - 0s 14ms/step
Ranking of reviews for your query: 193           "Hermione's intellect shines
throughout."
322              "Hermione's intelligence is inspiring."
289                 "The cinematography is outstanding."
393    "The film's tone is inconsistent and fails to …
375                       "The storyline is captivating."
                                 …
343    "Rubeus Hagrid's love for magical creatures ad…
137    "Rubeus Hagrid's love for magical creatures is…
5      "Rubeus Hagrid's love for magical creatures is…
265    "Rubeus Hagrid's love for magical creatures is…
333    "Rubeus Hagrid's love for magical creatures is…
Name: comment, Length: 491, dtype: object
```

Conclusions

- A DNN with a large set of parameters can easily overfit to smaller training datasets (Zhang et al., 2016). Therefore, during model design it is typical to strike a balance between the number of model parameters and the size of the data available for training.
  - In this case, K-fold cross-validation, keeping the model shallow (reducing the number of neurons) and iperparameter tuning helped the model to generalize.

- Some metrics could have been applied during the evaluation phase of the ranking task (as they have been after the classification task performed to test the model) if we had a query relevance labels for each comment.

- Other model architectures can fit on the specific problem, although CNNs are usually a good choice.

Kaggle.com

Manning, C. D., Raghavan, P., Schütze, H. (2008), “Introduction to Information Retrieval, Cambridge University Press, ISBN: 0521865719

Mitra, B., Craswell, N., (2018), “An Introduction to Neural Information Retrieval”, : Vol. xx, No. xx, pp 1–18. doi: 10.1561/XXXXXXXXX

Newatia, R. (2019), “How to implement CNN for NLP tasks like Sentence Classification”, medium.com/saarthi-ai/sentence-classification-using-convolutional-neural-networks-ddad72c7048c

Shen, Y.; He, X.; Gao, J.; Deng, L.; Mesnil, G. (2014) [ACM Press the 23rd ACM International Conference - Shanghai, China (2014.11.03-2014.11.07)] Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management - CIKM “14 - A Latent Semantic Model with Convolutional-Pooling Structure for Information Retrieval.”, 101–110. doi:10.1145/2661829.2661935

Zhang, Y., & Wallace, B. (2015), “A Sensitivity Analysis of (and Practitioners’ Guide to) Convolutional Neural Networks for Sentence Classification”