# Resultado de imagen de uocPEC 2: State of the art

Author: Teguayco Gutiérrez González

Studies: Master’s Degree in Data Science

## 2.1 Introduction

As explained in the introductory section, the problem of matching products coming from different sources brings with it the challenge of identifying identical products whose data may be missing (for instance, universal identifiers such as EAN or UPC codes could not be found) or presented in multiple formats. These difficulties make this matching task to be a non-trivial one and, as a result, a lot of research studies and tools have already arisen, most of them implementing Natural Language Processing (NLP) and Machine Learning techniques.

In this section, a comprehensive review of these previous works will be done based on the approaches they pose to analyse how they implement the solutions for this issue. It should also be noted that some of these works use more than one approaches.

## 2.2 Approaches

### 2.2.1 Text similarity

The first approach could consist in comparing two product titles or descriptions to determine their degree of similarity, as the same product offers coming from different web shops should have similar titles and descriptions. In NLP, a common solution is to use TF-IDF vectorization and calculate some measure, like the cosine similarity, to quantify the degree of similarity between two titles. TF-IDF vectorization could constitute a first valid approach, as it is a method which is able to identify the most important words in a given document.

However, in [1] they pose an adaptative string similarity measure that improves the matching results using TF-IDF and therefore demonstrate that the latter method has limitations when addressing the product matching problem. This proposal is based on giving higher weights to those terms in product titles that are more relevant to identify the product, as depending on the product, some terms will identify them better than others. For example, to identify a camcorder like the following:

**Canon Vixia HF S10**

The manufacturer code (HF S10) should have a high weight, as it is the most relevant term to identify the product. However, for the following cleaning toolkit for the former camcorder:

**Canon Vixia HF S10 Cleaning Toolkit**

The type of product (cleaning toolkit) should have the highest weight in order to avoid matching it with the former camcorder.

### 2.2.2 Attribute extraction

Most of the times, product titles include technical information which is crucial to identify them. However, in order to apply Machine Learning techniques, a more structured data about products should be managed. Therefore, the problem of obtaining these structured data about products from their textual information is arisen.

As an example, from the following product title:

# **Apple iPhone XS (64GB) – Gold**

It would be desired to filter it to obtain the following structured data:

{

“Brand”: “Apple”,

“ProductName”: “iPhone”,

“Model”: “XS”,

“MemoryCapacity”: “64”,

“MemoryCapacityUnit”: “GB”,

“Color”: “Gold”

}

This problem can be seen as an instance of an NLP task known as **Named-Entity Recognition** (NER) []. The goal in NER is to locate and classify the entities mentioned inside some given unstructured text.

#### 2.2.2.1 Simplest solution

One straightforward solution may consist in using **regular expressions** and **dictionaries** of known values for the attributes to extract. However, this approach can have several disadvantages.

On one hand, the use of regular expressions could lead to an inaccurate identification of attributes. For example, given the product name “Apple iPhone 32GB”, a regular expression can be written to extract the phrase “32GB”. However, this method would not be able to correctly resolve if this value correspond to RAM memory or to hard-disk capacity, as both attributes are commonly given in GB. Furthermore, as the information may be presented in many different formats, a regular expression should be written to handle every instance. For example, the inches of the screen for a certain TV could be written like “50-inches” or ‘50”’, so two different regular expressions are needed in this case.

On the other hand, the use of dictionaries may be a good approach for attributes with a fixed list of possible values like the colour. However, it could fail with attributes with an open list of possible values such as the brand, as new brands come eventually out on the market. In this latter case, the dictionary used to perform brand extraction should be regularly updated.

#### 2.2.2.2. Sequence labelling

Some studies like [2] and [3] address this issue by implementing more sophisticated methods. They first train some NER model to extract products features from their titles to then train Machine Learning models based on the structured data obtained to make the matching.

Thus, ***Conditional Random Fields*** (CRF) can be seen as the favourite alternative, as it is used by both of the above-cited works. It is a linear model for sequential labels. In fact, CRF can be considered as the sequential version of logistic regression.

Another implemented solution by [2] along with CRF is ***Structured Perceptron***, which is a supervised learning algorithm. By contrast, [3] goes beyond implementing CRF also with **text embeddings**.

The benefit of these sequence labelling algorithms is that they leverage the information given by the context of a given word in a certain product title. For example, in a product title like “Apple iPhone 4 (4GB RAM) - Black”, “4GB” could be recognized as RAM memory considering the next word is “RAM”. Using some of the above-mentioned simplest solutions, this information would not be exploited.

Nevertheless, sequence labelling algorithms need a significant sizable training set (usually labelled using BIO encoding) whose manual labelling would require a huge usage time. Some implemented alternatives to avoid this manual task consist in the use of “distant supervision”, in which a training dataset is automatically built based on heuristics and rules.

### 2.2.3 Image recognition

Matching products identifying these by performing image recognition on them could constitute the hardest approach of all. It is not only the complexity of models like ***Convolutional Neural Networks*** (CNN), which are a common solution for image recognition problems, but also some other challenges that have to be faced regarding products recognition from images: the same product can be found photographed from different perspectives, with different colours or different levels of brightness. Furthermore, CNNs could need a huge number of images to be trained, something that translates into managing and storing a lot of bytes of data.

Indeed, [3] implements also a CNN on the premise that most of the web shops use the same image for identical products. Treating the product matching problem as a two-class classification problem (given a pair of products, decide if they match or not), the implemented CNN allows to obtain image embeddings for the candidate products pair. Then, cosine similarity between both vectors is applied to determine if they are similar enough to match. However, this work consider similarity image embeddings as a weak indicator for product matching, as it performs better for the categorization prediction task.

## 2.3 References

1. Thor, A. (2010). Toward an adaptive string similarity measure for matching product offers. Informatik 2010: Service Science.
2. More, A. (2016). “Attribute Extraction from Product Titles in eCommerce”, arXiv preprint arXiv:1608.04670.
3. Petar Ristoskia, Petar Petrovskia, Peter Mikab, and Heiko Paulheima. (2017). A machine learning approach for product matching and categorization. Semantic web.