

# CheMapBERT Project Updates

Collaborators: *Anshumaan Chauhan*

[Github](#)

## 1 External Knowledge Source

Choice of External Knowledge base is extremely critical to observe a significant impact on the performance of the Machine Learning model. The easiest way for this could be scrape data using a Python script from several trustworthy sites as Wikipedia. The problem with this approach is that it could lead to our client getting blocked. Moreover, Wikipedia is way too large database, and to properly scrape it for our project, we do not need to scrape the entire Wikipedia. Although there is an export functionality that is offered by Wikipedia, that lets us export a Wikipedia page as XML format, but this technique requires us to know the pages that we want to export, gathering this information is highly inefficient. Therefore we move onto the use of existing formed databases, which offer the information required either by an API or the scraping from them is computationally possible.

- **DBPedia** : DBpedia is a database that contains the information present on Wikipedia in a structured format. This information can be easily queried using SPARQL (called "Sparkle"). The issue with using DBPedia as a knowledge source are:
  - There are a lot of products that are available with different composition of chemical ingredients which are not listed on Wikipedia, which means they are not available on DBPedia as well (Figure 1,2).
  - If we want to query where we need to use multiple dbps then we have to make use of UNION operator; and because we have many chemical ingredients that we will be using for the filtering of the web pages, UNION function for each one of them will make task inefficient and slow.
  - Lastly, DBPedia has a limit on the number of results returned for a given query (10000 rows per query).

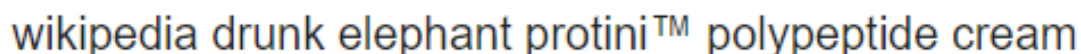


Figure 1: Wikipedia Search for a Chemical Product

- **INCIDecoder** : INCI is short form notation for International Nomenclature of Cosmetic Ingredients. INCI is a list of the standardized names to be used in the product ingredient description of any cosmetic/ skin care product and was published by the Personal Care Product Council. INCIDecoder is a site founded by Judit Rácz, and it used by people to know which ingredient is harmful/helpful for the skin. We are using this as an external database, due to the simple nature of the website and it is easy to scrape (no public API available). The list of ingredients can be passed with the URL request itself, and the results are different kinds of products that contains all the chemical ingredients.
- **Schema.org** : Schema markup is a structured data vocabulary that is used to better the understanding of the data on several websites. If we look at Google, it only understands 32 types of

About 6,760 results (1.04 seconds)

 Drunk Elephant  
<https://www.drunkelephant.com> › moisturizers › protini... ⋮

### Protini™ Polypeptide Cream | Strengthening Protein Face ...

**Drunk Elephant's** protein moisturizer combines signal peptides, growth factors, supportive amino acids, and pygmy waterlily for immediate improvement in the ...


★★★★★ Rating: 4.5 · 314 reviews · \$68.00 · 30-day returns · In stock

Missing: [wikipedia](#) | Must include: [wikipedia](#)

 Drunk Elephant  
<https://www.drunkelephant.com> ⋮

### Drunk Elephant | Biocompatible Skincare

A's for Everyone! For a limited time, receive a free deluxe sample of new retinol-powered A-Shaba Complex Eye Serum on all orders \$50+.

 Drunk Elephant  
<https://www.drunkelephant.com> › about-us ⋮

### Our Story in Founder Tiffany Masterson's Own Words

Read about how Tiffany Masterson started **Drunk Elephant** and our philosophy on creating innovative, effective skincare for all skin types in her own words.

Figure 2: Wikipedia Results

schemas, but not all websites fall within these schema markups, therefore Schema.org describes several other types of items, each has its own unique set of properties used for describing them. Issue with the Product schema used here was it does not capture the chemical ingredients used in the cosmetic products in the properties of the product.

- **Cosing** : Cosing is a database that is maintained by European Commission, and provides information about several chemical ingredients and the allowed composition of these ingredients in any product (along with their properties and uses). This database includes the information about cosmetic products, ingredients used in them and the type of cosmetic product it is. But unfortunately, this database is not available to public through any medium.
- **ChEBI** : ChEBI is a dictionary based database, that contains several components that are present inside each ingredient. However, it does not contain any information about how these ingredients are used in different combination in a product, and neither do they contain any information about any type of product.
- **EWG Skin Deep Database** : A database maintained by Environmental Working Group, and contains information about the chemical composition of a product along with the rating similar to INCIDecoder - based on how it affects the health. The issue with this database is that it can only be filtered based on one chemical ingredient and not multiple. The output it provides is quite helpful - type of product which contains this ingredient along with how many of such products is listed in the database. This information can be used in the form of a weighted average label (just like how we use attention mechanism in Transformers) if we are filtering the product based on only a single component present in it.
- **Open Product Data** : It is a database that contains information about all the commercialized products in the world. Unfortunately, the database is now archived by the Open Knowledge Foundation and can no longer be accessed by the public.
- **Open Beauty Facts Database** : It is a cosmetic product database, containing information about the product - such as its name, their chemical composition, the brand and a focus on whether the product is organic or not. Some drawbacks of this database is that it is still under development and therefore it has a lot of products mentioned, whose chemical ingredients are not mentioned in text but rather as an image. Secondly, this is a huge database that has no API that can be utilised for querying it over a set of ingredients - we have to download the whole database and query it using Python (if using csv file) or MongoDB, which is computationally intractable for us.

## 2 Model Fine-Tuning

Considering the previous limitations and drawbacks, we have implemented the following changes to overcome them.

- **More training epochs**: We have trained the Task Specific model for more number of epochs (15, previously was trained on 5 epochs) so that chemical mapping is learned more in a better fashion.
- **Merging of Labels**: Earlier the dataset was composed of 149 labels, which made it hard to learn a proper mapping during the training (very few samples per label). In the modified dataset, we have merged a group of labels into a single category so that model has enough samples for each label.

- **Data Shuffling:** Data shuffling has been proved to increase the performance of Machine Learning models in past, but it was not done for the Large Language Models, because of the huge amount of data they were pretrained on. In our case, the training dataset is very limited, and is grouped by the labels, therefore we have shuffled the data before we spilt the dataset into train and test.
- **Model Architecture:** BertForMaskedLM model was used with an addition of a classification layer (consisted of a single Linear Feed Forward layer), which resulted in a poor performance. We have now transitioned into a BertModel which is more suitable for the task of classification. BertModel was initialized with the default parameters, and then weights from Domain Specific BertForMaskedLM model were used to re-initialize the model. Architectural changes such as additions of Feed Forward and Dropout Layers on top of the Encoder block were made to make it a classification model. Now this updated BertModel was trained on the labeled dataset in a supervised learning paradigm for the task of chemical mapping of ingredients to a product label.

### 3 Knowledge Retrieval

#### 3.1 INCIDecoder

As discussed in Section 1, we are using INCIDecoder as our external knowledge source. INCIDecoder is a filtering based site, where we filter out products based on the chemical ingredients a product contains. In Figure 3 and 4, we demonstrate how to filter out some of the cosmetic products that contain "Saccharide Isomerate" in their chemical composition.

## Advanced Product Search Based on Ingredients

**WORDS OF PRODUCT NAME OR DESCRIPTION:**

**MUST CONTAIN ALL OF THESE INGREDIENTS:**

× Saccharide Isomerate

**MUST NOT CONTAIN ANY OF THESE INGREDIENTS:**

**SEARCH**

Figure 3: Querying/Filtering INCIDecoder Database

#### 3.2 Proposed Knowledge Extraction Methodology

In this section we have presented the proposed algorithm for the task of External Knowledge Retrieval from the Knowledge Source INCIDecoder (Figure 5).

cnk Christine Niklas AA Creme - Anti Redness  
Charlotte Tilbury Magic Eye Rescue Cream  
Code of Harmony Chill Gel Masque - Cbd Face Mask  
SVR Laboratoires Topialyse Palpebral Cc NYDG Ultra-Light Hydra-Gel  
Tata Harper Elixir Vitae Eye Serum Janssen Cosmetics Skin Contour Cream  
La Mer The Broad Spectrum Spf 50 Daily Uv Protecting Fluid  
Dermasence RosaMin Serum  
Susanne Kaufmann Nutrient Concentrate Skin Smoothing  
Derma E Essentials Radiance Toner  
Eisenberg Youth Elixir Illuminating Lifting Gel  
Dr. van der Hoog Hypoallergen Anti-Aging Day Cream Ida Warg Tanning Drops  
Philosophy Renewed Hope In A Jar Peeling Face Mask  
Tata Harper Hydrating Floral Essence DerMend Alpha + Beta Hydroxy Therapy  
Trader Joe's Ultra Hydrating Gel Moisturizer

Figure 4: Results of INCIDecoder Querying

1. Obtain the list of chemical ingredients as input. The input would be in the form of Natural Language, that is it will contain punctuation between the ingredients and may contain stopwords.
2. Input is then tokenized using a punctuation based tokenizer, that separates each ingredient as an individual token.
3. The individual tokens obtained are to be concatenated in the URL request sent to the INCIDecoder in a specialized format. Tokens with multiple words should be joined using an '+' operator in between them.
4. Send the URL as a GET request to the INCIDecoder using requests library.
5. INCIDecoder then uses the URL sent (containing the ingredients) for the Advanced Search of products that contain the ingredients and return a JSON file.
6. JSON file is parsed using BeautifulSoup library and the external information is extracted.

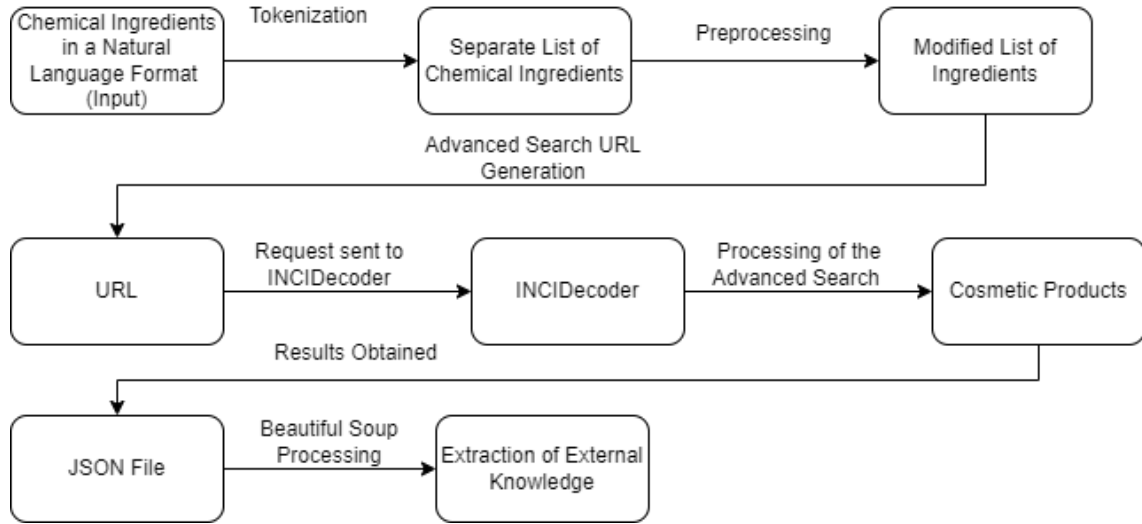


Figure 5: External Knowledge Extraction

## 4 Processing External Information

In the external information obtained, we have a list of items that contain the product type, along with product name, brand name and several other attributes that may not be useful to us (Figure 4). We want to test the effect of different preprocessing techniques on the performance of the Task Specific BERT model.