Horizon 2020

Understanding Europe's Fashion Data Universe

Named Entity Recognition and Linking Methods

Deliverable number: D2.1

Version 3.0

Funded by the European Union's Horizon 2020 research and innovation programme under Grant Agreement No. 699924

Project Acronym:	FashionBrain
Project Full Title:	Understanding Europe's Fashion Data Universe
Call:	H2020-ICT-2016-1
Topic:	ICT-14-2016-2017, Big Data PPP: Cross-sectorial and cross-lingual data integration and experimentation
Project URL:	https://fashionbrain-project.eu

Deliverable type	Report (R)
Dissemination level	Public (PU)
Contractual Delivery Date	30 June 2018
Actual Delivery Date	29 June 2019
Number of pages	24, the last one being no. 18
Authors	Ines Arous, Mourad Khayati - UNIFR
Peer review	Alessandro Checco - USFD Benjamin Winter - BEUTH

Change Log

Version	Date	Status	Partner	Remarks
0.1	01/06/2018	Draft	UNIFR	
1.0	30/06/2018	First submission	UNIFR & all	
2.0	30/12/2018	Revised first submission	UNIFR	
3.0	29/06/2019	Second submission	UNIFR & all	

Deliverable Description

As a result of task 1.1, this deliverable will consist of implemented algorithms for entity extraction from textual documents and linking to the taxonomy defined in WP1. The result will feed into WP4, 5, and 6.

Abstract

In this deliverable, we implement a new tool to process textual fashion dataset with noisy labels. We evaluate state of the art methods for named entity recognition and extend them with a bootstrap approach to improve their robustness against noisy labels. We integrate the best method for fashion items identification into our system and apply the latter to link the identified items to the FashionBrain taxonomy. In addition, our tool is able to enrich the FashionBrain taxonomy with newly discovered fashion items.

Table of Contents

List of Figures

List of Tables

List of Acronyms and Abbreviations

FBT	FashionBrain taxonomy
HKFCC	Fashion Communication Corpus of Hong Kong Polytechnic University
BiLSTM	Bidirectional Long Short-term Memory Networks
CRF	Conditional Random Field
NLP	Natural Language Processing
RNN	Recurrent Neural Network
NER	Named Entity Recognition
NEL	Named Entity Linking

1 Introduction

1.1 Motivation of the Deliverable

With the emergence of social media and fashion blogs, there has been an increasing interest in performing entity-centric analytics on the fashion domain. In fact, these blogs are a valuable source to spot new fashion items or new trends. In order to leverage the content of fashion blogs, we propose to build a new tool (named "FashionNLP) able to perform Natural Language Processing (NLP) on a fashion dataset making it possible to extract fashion entities and link them to the FashionBrain taxonomy (FBT). State of the art NLP systems [4, 13, 14] fall in short in providing such a tool as they rely on the redundant appearance of entities which does not allow to identify new fashion items.

FashionNLP is designed such that it identifies fashion items in a large textual corpus and links the fashion items to the FBT. In case the extracted item does not exist in the taxonomy, we add the new item to the most appropriate category in the FBT. One of the main components of the FashionNLP tool is the identification of fashion items. We evaluate state of the art methods for Named Entity Recognition (NER) on a large fashion dataset. This dataset contains noisy labels, i.e., some fashion items are not correctly identified, which lowers down the performance of any NER solution. Therefore, we propose to extend these NER techniques using a bootstrap approach in order to improve their robustness against noisy labels.

We use a large dataset retrieved from the Fashion Communication Corpus of Hong Kong Polytechnic University (HKFCC) [9]. HKFCC comprises 1 million words of English texts, grouped into five categories, including blogs (208,124 words), comments (205,248 words) and styling tips and product launches (204,71 words). In order to retrieve data from this corpus, we needed to provide specific queries.

1.2 Scope of the Deliverable

The result of this task will help us enrich the pre-built taxonomy and will input to WP 4 and WP 5.

This deliverable (D2.1) is part of WP2 in which we perform semantic data Integration for the fashion industry. In this deliverable, we focus on entity recognition and linking of fashion items. The relation extraction task was extensively covered in D4.3, therefore, in our deliverable, we provide a summary of the main outcomes of the deliverable D4.3.

D2.1 introduces a tool to identify fashion entities and link them to the FashionBrain taxonomy. The outcome of this deliverable will feed into D4.4 where extracted fashion items will be used for relation extraction. Also, D2.1 is strongly contributing to the research challenges about Brand Monitoring for Internal Stakeholders. Since our tool links to the FBT, it is contributing to Challenge 4: Linking Entities to Product Catalogue. In addition, the extracted fashion entities could be used for mining fashion reviews and textual data and hence it contributes to Challenge 5: Opinion Mining on Fashion Reviews and Challenge 6: OLAP Queries over Text and Catalogue Data. Moreover, D2.1 deals with Semantic Integration and hence contributes to the Core Technologies about Data Curation and Integration (i.e., CT1 Semantic Integration).

2 Background

In this section, we describe the NLP techniques used in this deliverable. More specifically, we describe the Named Entity Recognition techniques used to identify fashion entities, the Named Entity Linking technique used to match the identified entities to the FashionBrain taxonomy and summarize the Relation Extraction covered in D4.3.

2.1 Named Entity Recognition (NER)

Entity Recognition is the task of locating and classifying atomic elements in text into predefined categories [11]. In this step, we will locate key concepts inside textual data sources with noisy labels such as fashion blogs and classify them as fashion or non fashion concepts. We implement three approaches for NER: 1) GloVe [6], is an unsupervised learning method that captures global corpus statistics. 2) Character-based embedding [8], is a BiLSTM-CRF model that uses a "character-based" representation of words. 3) Flair [1], is also a BiLSTM-CRF model that uses a contextual string embedding to represent words as vectors. We first provide a brief description of LSTMs and CRFs, then we describe each one of the aforementioned method as a word embedding technique.

2.1.1 BiLSTM-CRF

Current state-of-the-art approaches for NER tasks typically use the BiLSTM-CRF model. The Bidirectional Long Short-term Memory Networks (BiLSTM) is a variant of a bidirectional recurrent neural networks. Typically, the BiLSTM network is used with a Conditional Random Field (CRF) decoding layer. In what follows, we provide more details about these two concepts:

- Bi-LSTM (Bidirectional Long Short-term Memory Networks): Long Short-term Memory Network is an artifical Recurrent Neural Network (RNN) with feedback connections. This network takes as input a sequence of vectors $(x_1, x_2, ..., x_n)$ and returns another sequence $(h_1, h_2, ..., h_n)$. BiLSTM incorporats a memory-cell which allows it to captures long-range dependencies contrary to regular RNN that are biased towards their most recent inputs in the sequence. For a given sentence $(x_1, x_2, ..., x_n)$ containing n words, each word is represented as a d-dimensional vector. We use one BiLSTM to compute the representation of each word from the left context and another BiLSTM that reads the same sequence of words in reverse to give the representation

of each word from the right context. These two LSTMs paired together are referred to as a bidirectional LSTM. Using a bidirectional LSTM allow us to effectively include a representation of a word in context, which is useful for tagging fashion items.

- CRF (Conditional Random Field): Instead of tagging a sentence as a sequence of words independent from each other, we model the words in a sentence jointly using a CRF [7]. Specifically, for an input sentence $\mathbf{X} = (x_1, x_2, ..., x_n)$ containing n words, we want to obtain as output a sequence of tags $\mathbf{Y} = (y_1, y_2, ..., y_n)$. Using CRF, we compute the score $s(\mathbf{X}, \mathbf{Y})$ of all possible sequences. This score takes into account two parameters: 1) $\mathbf{P}$, a matrix of scores outputed by a bidirectional LSTM, where each element $\mathbf{P}_{i,j}$ corresponds to the score of the j-th tag of the i-th word in a sentence. 2) $\mathbf{A}$ is a matrix of transition scores such that $\mathbf{A}_{i,j}$ represents the score of a transition from the tag i to tag j. The details about the score computation is out of the scope of this report. We refer the readers to Section 2.2 of [8] for more details.

2.1.2 Word Embedding

State of the art technique for NER differ essentially in the method used for "word embedding". In what follows, we present three state of the art NER technique and briefly describe their "word embedding" method:

- Global Vectors for Word Representation (GloVe): "GloVe" was proposed by Pennington et al. in [10]. GloVe is an unsupervised learning algorithm for obtaining vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space.
- Neural Architectures for Named Entity Recognition (Char-embedding): This model was introduced in [8] by Lample et al. for NER tasks. In their model, they use a character-based representation of words which has the advantage of learning representations specific to the task and domain at hand.
- Contextual String Embeddings for Sequence Labeling (Flair): In [1], the authors propose a model named "Flair". This model captures word meaning in context and therefore produce different embeddings for polysemous words depending on their usage. In addition, Flair models words and context fundamentally as sequences of characters, to both better handle rare and misspelled words as well as model sub-word structures such as prefixes and endings.

2.2 Named Entity Linking and Taxonomy Enrichment

Entity Linking is the task of assigning entities from a textual mentions of such entities in a document to a Knowledge Base [5]. In this section, we explain first, how to

link the fashion concepts to FashionBrain taxonomy and then, in case the extracted concept does not exist in the taxonomy, we will add it to the most appropriate category in the FBT.

2.2.1 Named Entity Linking

An intuitive approach to perform Named Entity Linking (NEL) consists in applying a string matching approach. We use the Damerau-Levenstein distance to measure the distance between the identified entity with an NER model and the FBT concepts. The Levenstein distance between two strings is defined as the minimal number of edits required to convert one into the other. We use the Levenstein ratio which is derived from the Levenstein distance and varies between 0 and 1 where a value equals to 1 represents no required edits. The Levenstein ratio r between two strings a and b is defined as follows:

$$r = 1 - \frac{n}{(size(a) + size(b))}$$

We allow the so called "fuzzy" matching where we allow misspelled text to also be linked to the FBT.

2.2.2 Taxonomy Enrichment

Using the BiLSTM-CRF approach, we are able to extract not only the fashion items existing in the FBT but also new fashion items. In fact, the BiLSTM-CRF learns from the context the position of words representing fashion items and hence, it is able to identify fashion items that do not exist in the FBT. We have two cases with the new discovered items:

- Case 1: The discovered item is a compound noun where the last word of this compound noun is a class c in the FBT. This is a very common case and we proceed by adding this discovered item as a subclass of c in the FBT.
- Case 2: The discovered item has no common words with the items existing in the FBT. In this case, we seek help from the crowd by designing a task where they classify the discovered item in the FBT.

2.3 Relation Extraction

Relation Extraction (RE) is the task of extracting and classifying relations between entities in unstructured data like texts. Since this RE is already covered in D4.3, we decided to omit this task from our deliverable and include a summary of the main outcomes.

In D4.3, three different relation extraction approaches were evaluated on three datasets: 1)OpenIE [2] applies generic lexico-syntactic patterns to detect un-typed relation candidates. 2) HRL-RE [12] is based on hierarchical reinforcement learning and BiLSTM. This model solves both the task of binary relation extraction and the task of NER. 3)SECTOR [3], a new model proposed in this deliverable, is a semi-supervised method and is able to extract relations in long paragraphs. These approaches have a strong dependency on the properties of the dataset used: If the dataset contains long paragraphs with redundant types of relation, then in this case, SECTOR has the best performance. If the dataset is large, then HRL-RE is the best model to use. If the dataset is unlabeled, then the only approach that can be used in such case is OpenIE. The dataset HKFCC [9] used in our deliverable (D2.1) is not yet labeled for the relation extraction task. Extensive work is being conducted in order to utilize crowd workers to annotate both the entities and the relation types. The application of the developed method SECTOR on a fashion dataset will be covered in D4.4.

3 Tool

In this section, we describe our new natural language processing tool called FashionNLP which is specially designed for fashion textual data. This tool extends existing state of the art NER techniques to fashion applications. More specifically, FashionNLP has three main components: NER, where fashion entities are recognized on textual data, NEL, where we link the fashion entity to the FashionBrain taxonomy and finally, taxonomy enrichment, where non-existing fashion entity will be added to our FashionBrain taxonomy.

This section is organized as follows: First, we explain in detail the FashionNLP pipeline. Second, we evaluate our tool using state of the art techniques. Finally, we provide a quick start guide for the tool explaining how to install it and run it.

3.1 Pipeline

We first start with the Named Entity Recognition (NER) step. In this step, we split the data into three sets: training, validation and testing sets. We use for the training set the annotated fashion items from the processed data as it was described in Section 3.2 and we train the models introduced in Section 2.1. As a result of this task, we identify the fashion entities in the testing set. A pre-step of this identification is the tokenization step, i.e. each word in the corpus is tagged as a noun, a verb, an adjective or a date. We illustrate the NER step with an example.

Example 1. We process the following sentence with the trained NER model *"For Summer 2013 Supreme presents a small Wackies capsule collection , consisting of four graphic T-shirts, a tank top and a snapback cap"* and obtain the following as a result of the tagging step:

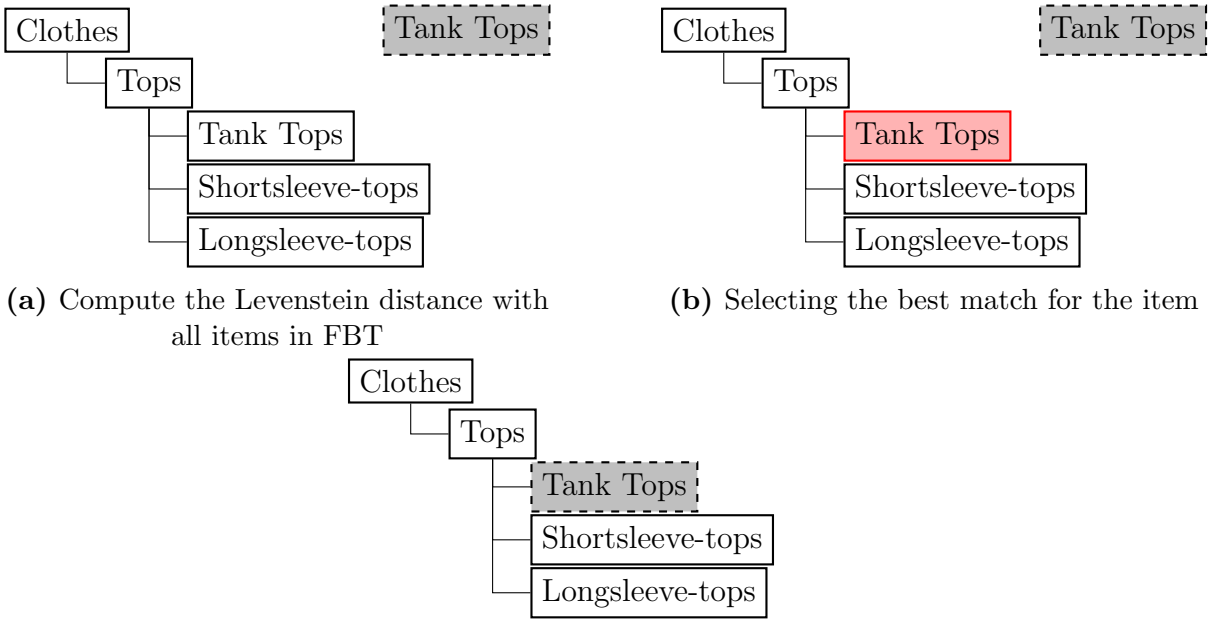
For_IN Summer_NNP 2013_CD, Supreme_NNP presents_VBZ a_DT small_JJ Wackies_NNP capsule_NN collection_NN, consisting_VBG of_IN four_CD graphic_JJ T-shirts_NNS, a_DT tank_NN top_NN, and_CC a_DT snapback_NN cap_NN.

Then the entities relevant to the fashion domain are extracted where we use two labels: the label 'O' means the entity is not relevant to fashion and 'ITEM' means the entity is a fashion item.

('For', 'O'), ('Summer', 'O'), ('2013', 'O'), ('Supreme', 'O'), ('presents', 'O'), ('a', 'O'), ('small', 'O'), ('Wackies', 'O'), ('capsule', 'O'), ('collection', 'O'), (',', 'O'), ('consisting', 'O'), ('of', 'O'), ('four', 'O'), ('graphic', 'ITEM'), ('T-shirts', 'ITEM'), (',', 'O'), ('a', 'O'), ('tank', 'ITEM'), ('top', 'ITEM'), (',', 'O'), ('and', 'O'), ('a', 'O'), ('snapback', 'ITEM'), ('cap', 'ITEM')

Here, the NER model identified “graphic T-shirts”, “tank top” and “snapback cap” as fashion items.

We do both the Named Entity Linking and the taxonomy enrichment steps simultaneously. We proceed by measuring the Levenstein distance between the extracted fashion items and the items in the FBT. In case the item exists, FashionNLP provides its best match in the FBT and link it to its parents as illustrated in Figure 3.1. If the item is a newly discovered item, FashionNLP provides a potential parent to the new item as illustrated in Figure 3.2.

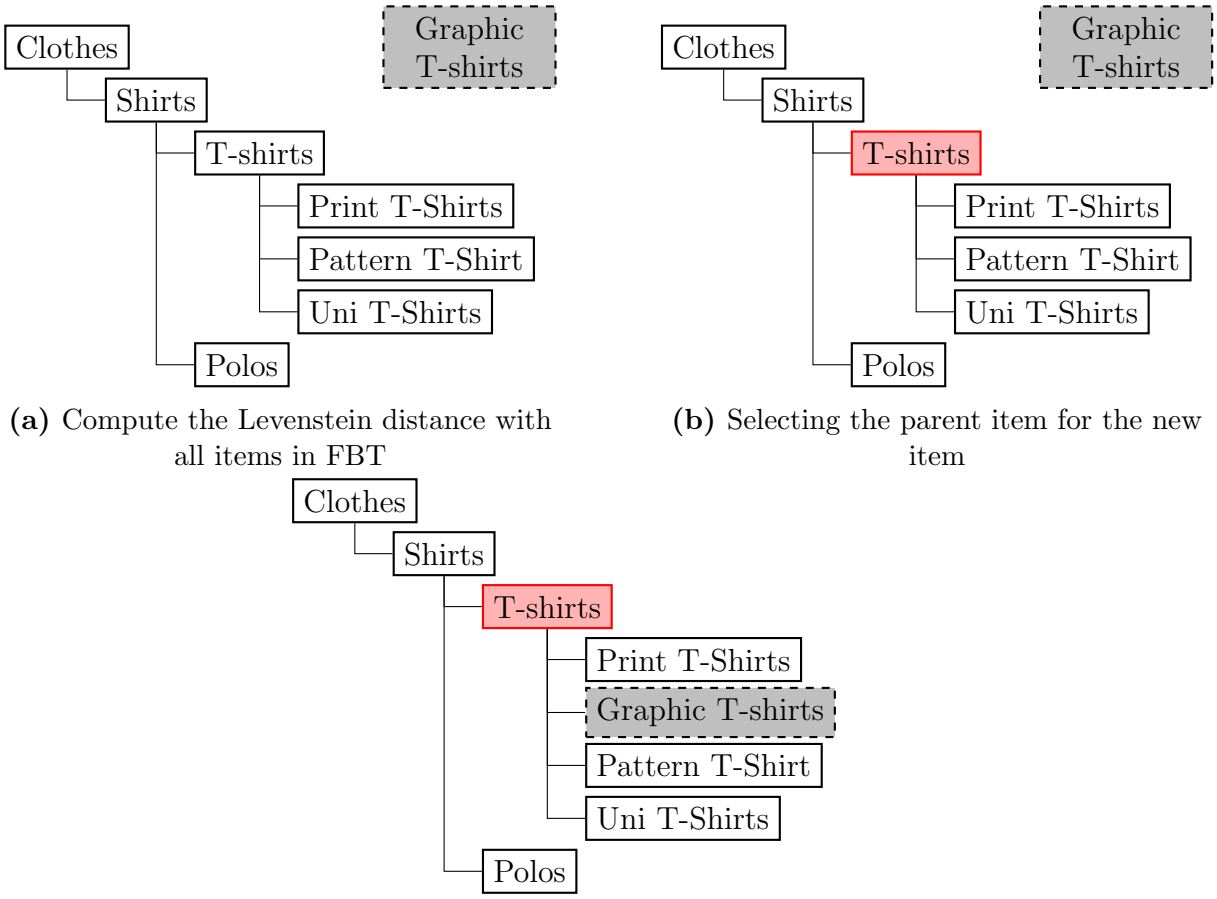


(a) Compute the Levenstein distance with all items in FBT

(b) Selecting the best match for the item

(c) Link the item to its best match

Figure 3.1: Example of Named Entity Linking



(a) Compute the Levenstein distance with all items in FBT

(b) Selecting the parent item for the new item

(c) Adding the new item to the FBT

Figure 3.2: Example of Taxonomy Enrichment

3.2 Dataset Labeling

In order to retrieve data from the HKFCC corpus [9], we needed to provide specific queries.

Example 2. *By entering the query "Fashion", the corpus returned 5,151 matches in 1,156 different texts. A match is given as follows: "Today was the final day of <<fashion >>month.", where the symbols "<<...>>" indicate the query tags.*

The fashion entities in this corpus are initially unlabeled. Thus, we use the query tags to annotate the data. In order to retrieve all relevant fashion entities, we provided as a query all fashion items from the FBT. Using this query, we obtained 77673 matches resulting roughly in 388,365 lines with annotated fashion items. This retrieved data contained some issues that can be summarized in the following points:

1. Redundant data: Sentences containing n fashion items are repeated n times where each time only one fashion item is tagged.

2. Irrelevant highlighted words: Some fashion items have a different meaning if used in an other context. These words were highlighted independently from the context.
   Example 3. *In both sentences: "I'd probably pair this top with colored trousers" and "A top designer could give a popular label an edge.", the word "top" was highlighted; while only in the first sentence, the word "top" refers to a fashion item.*
3. Unidentified fashion items: For example, the fashion items exiting in FBT as a compound nouns (e.g. summer dress) were not identified.

To solve the first and second problem, we proceed as follows: First, we eliminate all duplicates and make sure to annotate all fashion items existing in one sentence. Second, we use the Levenstein distance to compare between the fashion items in the FBT and the annotated words to eliminate all non relevant entities. Finally for the tagged queries that represent fashion entities only in specific context, we proceed by manually annotating them. Using these steps, we obtain 41k lines with annotated fashion items.

The last problem require to manually annotate all the data. However due to the large size of the dataset, we propose an alternative approach which consists in using a bootstrap approach to discover new fashion entities as described in the following Section 3.3.2.

3.3 Experiments

This section presents the experimental results for the evaluation of NER methods. First, we define the evaluation metrics that we use. Then, we introduce the bootstrap approach used to compare the performance of the NER methods and analyse the results.

3.3.1 Evaluation Metrics

In order to evaluate our tool, we use three metrics commonly used for NER and NEL tasks which are Precision, Recall and F1. These metrics are defined as follows:

$$\begin{aligned} precision &= \frac{tp}{tp + fp} \\ recall &= \frac{tp}{tp + fn} \\ F1 &= 2 \times \frac{precision \cdot recall}{precision + recall} \end{aligned} \tag{3.1}$$

where tp, fp and fn represent respectively true positive, false positive and false negative.

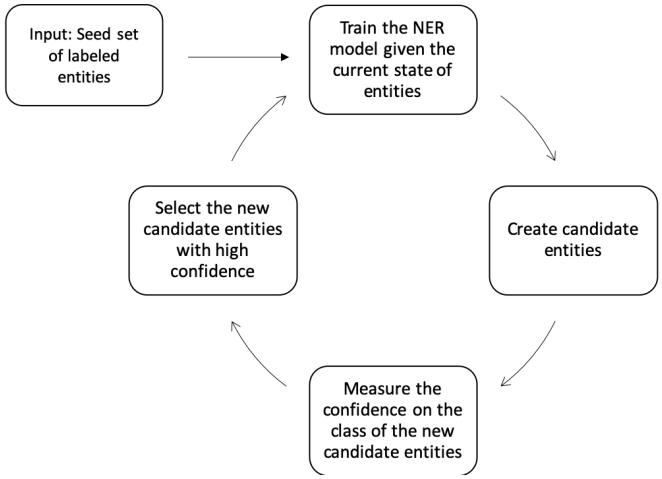


Figure 3.3: Bootstrap Approach

3.3.2 Results

As the data contains noisy labels, i.e. some of the fashion items in the training data are not identified, we propose to use a bootstrap approach on the three NER models introduced in Section 2.1. The bootstrap approach consists in the following steps: We use the annotated data from the data processing as a seed set of labeled entities to first train a model and then, we use the trained model to relearn the classes of the training set. Based on the confidence measure [1] obtained of the new candidate entities, we select those with high confidence measure. We repeat the aforementioned steps as it is illustrated in Figure 3.3.

Table 3.1 summarizes the performance of the three models: GloVe, Char-embedding and Flair on the fashion corpus using the bootstrap approach. Several interesting findings are obtained.

First, we observe that Char-embedding generally outperforms GloVe. Recall that Char-embedding uses the concatenation of the forward and backward representations of a word from the bidirectional LSTM while the GloVe embedding captures the global context based on the statistics of word occurrences in a corpus. Therefore, compared to the GloVe embedding, the character-based embedding brings the advantage of learning representations specific to the task and domain at hand.

Second, we observe that both models, GloVe and Char-embedding, have a degrading performance when using the bootstrap approach. In fact, both models achieve their

[1] The confidence measure is the probability estimated by a model that a word is an entity of a given type

best performance in the first iteration. In the third iteration, the F1-score of both GloVe and Char-embedding drop, with 4% and 0.36% respectively, compared to the first iteration. For both models, the bootstrap approach creates more entities labeled as 'Item' while it should be classified as 'O', therefore the number of true positives decreases and hence the result.

Most importantly, Flair achieves the best performance among all NER methods: it outperforms GloVe by 7% and Char-embedding by 3.5% in terms of F1. Moreover, the bootstrap approach has a positive effect on Flair where the F1-score increases by 0.1% in the second iteration and by 0.3% in the third iteration. This significant improvement clearly demonstrates the effectiveness of Flair in identifying fashion items in a large corpus.

Table 3.1: Evaluation of FashionNLP

Method		Precision	Recall	F1
GloVe [10]	1st iteration	0.8485	0.8604	**0.8544**
	2nd iteration	0.8299	0.8488	0.8392
	3rd iteration	0.8088	0.8326	0.8205
Char-embedding [8]	1st iteration	0.8812	0.8867	**0.8839**
	2nd iteration	0.8791	0.8879	0.8835
	3rd iteration	0.8766	0.8848	0.8807
FLAIR [1]	1st iteration	0.9123	0.918	0.9151
	2nd iteration	0.9132	0.9189	0.916
	3rd iteration	0.9148	0.9211	**0.9179**

3.3.3 Analysis

In order to have a better insight on the results obtained in Section 3.3.2, we analyse the impact of the bootstrap approach on the "confidence score". The confidence score is defined as the probability estimated by a model that a token is an entity of a given type. We measure the confidence score of the three models introduced in Section 2.1 in the first and the last bootstrap iteration and report the results respectively in Table 3.2.

Table 3.2: Average Confidence of NER models on the assigned classification on the first and last iteration

Method	GloVe		Char-embedding		Flair	
Iteration	1st	3rd	1st	3rd	1st	3rd
Confidence on Identified Items	0.973	0.979	0.914	0.981	0.978	0.984
Confidence on Correct Items	0.957	0.960	0.955	0.964	0.961	0.970

We observe that for all models, the confidence on the correct identified items increases: For Glove, the confidence increases by 0.31%; for Char-embedding, the confidence increases by 0.94% while for Flair, it increases by 0.61%. The number of correctly identified items is 4032 for all methods in both iterations except for Char-embedding, where in the first iteration, it is only able to identify correctly 3797 fashion items (see Table 3.3). The improvement in the confidence score for all models using the bootstrap approach confirms the impact of this approach on improving the robustness of these models.

Table 3.3: Average Confidence of NER models on the assigned classification on the last iteration

Method	GloVe		Char-embedding		Flair	
Iteration	1st	3rd	1st	3rd	1st	3rd
# Identified Items	4037	4037	4069	4037	4037	4037
# Correct Items	4032	4032	3797	4032	4032	4032

3.4 Code

Code is available through this link: https://github.com/eXascaleInfolab/fashion_nlp_v2

3.4.1 Getting Started

To install the FashionNlP tool, run the following command:

```
git clone https://github.com/FashionBrainTeam/fashion_nlp_v2
```

The "fashion_nlp" package contains three folders

- src: contains three python scripts
  - lstm_fashion.py: the implementation of the LSTM-CRF models
  - bootsrap_lstm.py: the implementation of the bootstraping approach
  - taxonomy_matching.py: the implementation of the taxonomy enrichment
- Data contains the training set (fashion_items_train.txt), the testing set (fashion_items_test.txt) and the FashionBrain taxonomy (FBtaxonomy.csv)
- output contains the results of the bootstrap approach.

3.4.2 Running the code

- The arguments needed to train an LSTM-CRF model are:

  - data_folder: Path to input folder containing training and testing sets
  - embedding: Type of embedding and it could be one of the three options: 'no_char', 'char' or 'flair'
  - epochs: Number of epochs to train the chosen model
  - output_folder: Path to the output folder to save three files: loss.tsv contains the accuracy measures in each epoch, test.tsv contains the the testing set with model labels and training.log contains the log history.

Example 4. *Running an experiment of LSTM-CRF model with the following parameters:*

  - *data_folder: '../data/lstm_input'*
  - *embedding: 'no_char'*
  - *epochs: 150*
  - *output_folder: '../output'*

```
python lstm_fashion.py --data_folder '../data/lstm_input'
--embedding 'no_char' --epochs 150 --output_folder
'../output'
```

- The arguments needed to use the bootstrap approach are:
  - model: Path to folder containing the model to load
  - first_iteration: Path to folder containing the first iteration input data
  - second_iteration: Path to folder containing the second iteration input data
  - epochs: Number of epochs to train the chosen model
  - retrained: Path to the data file trained in the second iteration
  - output_folder: Path to the output folder to save three files

Example 5. *Running an experiment of the bootstrap approach with the following parameters:*

  - *model: Path to folder containing the model to load*
  - *first_iteration: Path to folder containing the first iteration input data*
  - *second_iteration: Path to folder containing the second iteration input data*
  - *epochs: Number of epochs to train the chosen model*
  - *retrained: Path to the data file trained in the second iteration*
  - *output_folder: Path to the output folder to save three files*

```
python bootstrap_lstm.py
--model '../output/1st_iter/final-model.pt'
--first_iteration '../data/lstm_input' --second_iteration
'../data/lstm_bootstrap' --epochs 100 --retrained
'../data/lstm_bootstrap/retrained_data.tsv' --output_folder
```

```
'../output/2nd_iter'
```

- The arguments needed to enrich the FashionBrain taxonomy:
  - taxonomy: Path to the FashionBrain taxonomy
  - test_result: Path to the file containing the testing result

```
python taxonomy_matching.py
--taxonomy '../data/enrichment_input/FBtaxonomy.csv'
--test_result '../data/enrichment_input/test_result.txt'
```

3.4.3 Future Work

These results can be further improved with human in the loop approach to reduce the noisy labels in our dataset. The crowd can be involved in the taxonomy enrichment part as well: First, we can ask the crowd to place new fashion entities in the FBT and then, we can compare worker's answers to our tool output, i.e. the suggested parent for new items, and rectify the results.

The goal would be to build a model that imitates workers reasoning in placing new fashion items in the FBT.

4 Conclusions

In this deliverable, we have introduced a new natural language processing tool called FashionNLP, which is specially designed for fashion textual data. This tool uses FLAIR to extract fashion entities in textual data. FashionNLP performs three steps namely i) named entity recognition, ii) named entity linking and iii) taxonomy enrichment. The results we have obtained are promising and improve the state of the art.

These results can be further improved with human in the loop approach to reduce the noisy labels in our dataset, and to improve the taxonomy enrichment process.

Bibliography

[1] Alan Akbik, Duncan Blythe, and Roland Vollgraf. Contextual string embeddings for sequence labeling. In *COLING 2018, 27th International Conference on Computational Linguistics*, pages 1638–1649, 2018.

[2] Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D Manning. Leveraging linguistic structure for open domain information extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 344–354, 2015.

[3] Sebastian Arnold, Rudolf Schneider, Felix A Gers, Philippe Cudré-Mauroux, and Alexander Löser. SECTOR: A Neural Model for Joint Segmentation and Topic Classification. *TACL*, (to appear):12, 2019. 00000.

[4] Steffen Eger, Gözde Gül Sahin, Andreas Rücklé, Ji-Ung Lee, Claudia Schulz, Mohsen Mesgar, Krishnkant Swarnkar, Edwin Simpson, and Iryna Gurevych. Text processing like humans do: Visually attacking and shielding nlp systems. *arXiv preprint arXiv:1903.11508*, 2019.

[5] Zhaochen Guo and Denilson Barbosa. Robust entity linking via random walks. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 499–508. ACM, 2014.

[6] Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*, 2015.

[7] John Lafferty, Andrew McCallum, and Fernando CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. 2001.

[8] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*, 2016.

[9] The Language and The Hong Kong Polytechnic University Mutilmodal Analysis Lab(LAMAL), Department of English. Cqp web for language corpora. http://lamalcorpora.engl.polyu.edu.hk/cqpweb/, 2017. Accessed: 2019-06-13.

[10] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[11] Satoshi Sekine and Elisabete Ranchhod. *Named entities: recognition, classification and use*, volume 19. John Benjamins Publishing, 2009.

[12] Ryuichi Takanobu, Tianyang Zhang, Jiexi Liu, and Minlie Huang. A hierarchical framework for relation extraction with reinforcement learning. *CoRR*, abs/1811.03925, 2018. URL http://arxiv.org/abs/1811.03925.

[13] Charu Virmani, Dimple Juneja, and Anuradha Pillai. Design of query processing system to retrieve information from social network using nlp. *KSII Transactions on Internet & Information Systems*, 12(3), 2018.

[14] Daniel Zeman, Jan Hajič, Martin Popel, Martin Potthast, Milan Straka, Filip Ginter, Joakim Nivre, and Slav Petrov. Conll 2018 shared task: Multilingual parsing from raw text to universal dependencies. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–21, 2018.