

# A Hierarchical Deep Learning Natural Language Parser for Fashion

José Marcelino  
Farfetch  
Universidade de Coimbra  
jose.marcelino@farfetch.com

Luís Baía  
Farfetch  
luis.baia@farfetch.com

João Faria  
Farfetch  
joao.faria@farfetch.com

Ricardo Gamelas Sousa  
Farfetch  
ricardo.sousa@farfetch.com

## ABSTRACT

This work presents a hierarchical deep learning natural language parser for fashion. Our proposal intends not only to recognize fashion-domain entities but also to expose syntactic and morphologic insights. We leverage the usage of an architecture of specialist models, each one for a different task (from parsing to entity recognition). Such architecture renders a hierarchical model able to capture the nuances of the fashion language. The natural language parser is able to deal with textual ambiguities which are left unresolved by our currently existing solution. Our empirical results establish a robust baseline, which justifies the use of hierarchical architectures of deep learning models while opening new research avenues to explore.

## CCS CONCEPTS

• **Computing methodologies** → **Natural Language Processing**; *Information extraction*; • **Information systems** → *Information retrieval*;

## KEYWORDS

Natural Language Processing, Information Extraction, Information Retrieval, Fashion e-commerce, Deep Learning

### ACM Reference Format:

José Marcelino, João Faria, Luís Baía, and Ricardo Gamelas Sousa. 2018. A Hierarchical Deep Learning Natural Language Parser for Fashion. In *Proceedings of KDD (KDD Workshop on AI for Fashion)*. ACM, New York, NY, USA, 7 pages. <https://doi.org/10.1145/nnnnnnn.nnnnnnn>

## 1 INTRODUCTION

Natural Language Processing (NLP), despite being a vast domain with numerous dedicated works, continues to present itself as a challenge. The recent proliferation of novel deep neural networks algorithms and methods paved the way to new capabilities in solving NLP tasks [8, 12].

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

*KDD Workshop on AI for Fashion, 2018, London, UK*  
© 2018 Copyright held by the owner/author(s).  
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.  
<https://doi.org/10.1145/nnnnnnn.nnnnnnn>

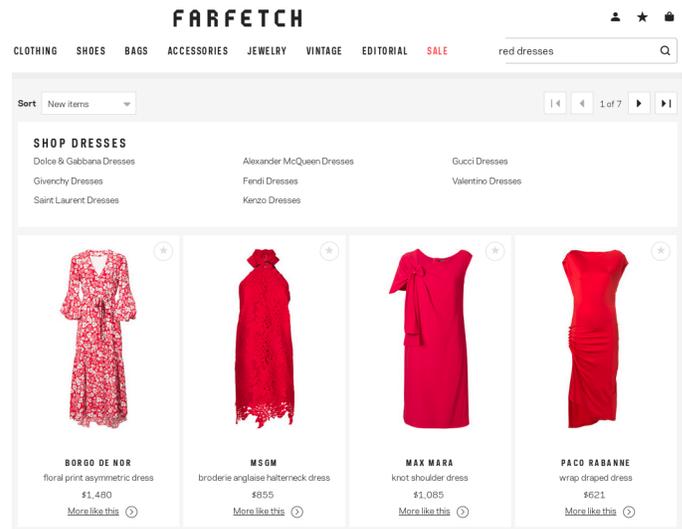


Figure 1: Farfetch search engine results for “red dresses” queries.

At Farfetch, the global platform for luxury, a robust search engine is required to cope with its users’ needs and expectations (Figure 1). Historically, the search engine parser has been keyword-based, involving a large number of constraint rules. As Farfetch expands, new ways to provide accurate search answers are required.

Though robust for general purposes, the limitations of the established keyword-based approach are twofold: 1) the inability to capture nuances of fashion epithets (for instance, semantically speaking, what is a floral dress?), and 2) an algorithm able to translate the domain into an effective identification of fashion entities in potentially ambiguous contexts.

The contributions of this work are the following:

1) we propose a hierarchical deep neural network parser for the fashion domain. Each of the modules of the deep neural network is specialized in a particular task: Word representation, Part-of-Speech tagging (PoS), Dependency Parsing (DP) and Named Entity Recognition (NER) — see Figure 2;

2) a transfer learning approach, to cope with the semantics of the fashion domain while reusing information from generic domains. We show the robustness of this network in a fashion dataset for

each of the different modules that compose the aforementioned architecture.

This work is structured as follows: In Section 2, we briefly present the current state-of-the-art. In Section 3, our proposal is thoroughly described followed by our preliminary results (Section 4). Conclusions are drawn in Section 5.

## 2 RELATED WORK

Behind each search engine, there is a drive to present relevant results by taking users' needs and expectations into account. Early works looked at query-document matching algorithm [13, 18] solutions. Although useful, they limit the interpretation and do not take context into account, resulting in models which fail in ambiguity prone environments. Recent advances in the literature lead us to the use of neural networks, providing us with an ability to detect complex non-linear patterns between sets of inputs. Although not explicitly, it also gives us the ability to leverage together several NLP tasks in a natural way. Transfer learning domain explores the reuse of pre-trained models and subsequent application to a different set of problems. Inspired by transfer learning, the appearance of holistic architectures to respond to complex NLP tasks [5, 6] started to emerge, either with many-task solutions [3, 6, 9] or simply end-to-end [4, 12] architectures.

The quest to a simple unified model, which tries to leverage on multi problems insights in a hierarchical way drives our path. The word representation problem explores embeddings techniques in order to map words in a vectorial space while being able to capture semantic similarities. Word embedding solutions presented in literature such as word2vec [15] or GloVe [17] establish a starting point towards our objective. Although new solutions that explore character embedding techniques emerged, such as fastText [11], we decided to delay their exploration to a later stage. To provide syntactic insights, Part-of-Speech tagging solutions took a leap forward from hidden markov models trends [1], with a use of deep end-to-end sequence labelling techniques [12]. The task of associating dependencies between words, Dependency Parsing, evolved into a set of two different approaches. Transition-based architectures, based in Chen [3] and Dyer [7] work, and graph-based architectures [16] dominate the state-of-the-art solutions to enrich with syntactic association information.

A vastly explored problem, Named Entity Recognition, deals with the identification of entities in textual content. In a similar fashion to the Part-of-Speech tagging task, end-to-end sequence labelling architectures [4, 12] dominate recent approach paths.

In Section 3 we will delve into the details of our proposal, a hierarchical Natural Language Parser.

## 3 SEMANTIC SEARCH

Common approaches for information retrieval encompass keyword matching methodologies. Although pragmatic, a caveat of such approaches is that the vast majority rely on rules, hence not being flexible enough or being hard to define.

At first sight, we can identify a few major challenges. Search queries follow a long tail distribution, with several different queries with extremely low frequency. Another problem is related to the semantic gap between queries and documents. The language can

also be different depending on which country Farfetch website is accessed. Finally, our vocabulary is also domain specific, making it necessary to develop a tailor-made fashion aware solution.

At this moment, Farfetch's search is mainly keyword-based and has lots of constraining rules. It lacks ability to interpret text, often creating noise due to unsolved ambiguities. The Natural Language Parser, which goes by the name of Semantic Search, provides a novel way to interpret text. Given a query, such as "*I want red dresses*", ideally, the Natural Language Parser should be able to extract the tag "*Colour*" from the word "*red*" and "*Category*" from the word "*dresses*". We increase the search engine robustness, giving it the ability to interpret more complex and semantically rich queries. Conceptually, the Natural Language Parser is sub-divided in a set of vital steps, namely Word Representation, Part-of-Speech tagging, Dependency Parsing and Named Entity Recognition. Each task has a hierarchical order, forming a pipeline which extracts, at each step, more information contained in the query.

Semantic Search's Word Representation module starts by slicing each sentence into a sequence of tokens, taking into consideration a text normalization step. We produce an embedding layer on top of the sequence of tokens, which maps words used in similar contexts closer together. In the next step, a Part-of-Speech tagging solution enriches content with sequence labelling methods. This identifies components of a given sentence, such as nouns, adjectives or verbs. Next in line, a Dependency Parser extracts relationships between words and resolves ambiguities concerning inheritance. The final module in the Semantic Search hierarchical pipeline extracts Named Entities, being responsible for identifying key tags (e.g Brand, Colour, Category, Materials) which are part of any fashion domain.

### 3.1 Word Representation

The Word Representation module is not only responsible for the word embedding phase but also for the pre-processing concerning normalization. It starts by slicing each sentence in a sequence of tokens, taking spaces and lower-case capitalization rules into consideration. Although common in literature, stemming, lemmatization and other more complex operations were not considered. The inherent simplification associated with those techniques would reduce the information content, with potential losses in terms of syntactical context.

The creation of a dense vector representation can be achieved through several different techniques. In our case we consider GloVe [17] and word2vec [15]. Both are word-based, taking into consideration the frequency of proximal words to optimize their numeric representation. Words that are used in the same context get mapped closer in the vectorial space.

Each token is associated with an embedded dense vector representation, which are specially prepared to deal with the fashion domain vocabulary, due to its text normalization and training nuances. Figure 2 section [A] illustrates the module architecture. More about the experimental study on Section 4.

### 3.2 Part-of-Speech Tagging

One of the pivotal tasks in our Natural Language Parser is the identification of components of a given sentence (e.g., noun, adverb, verb).

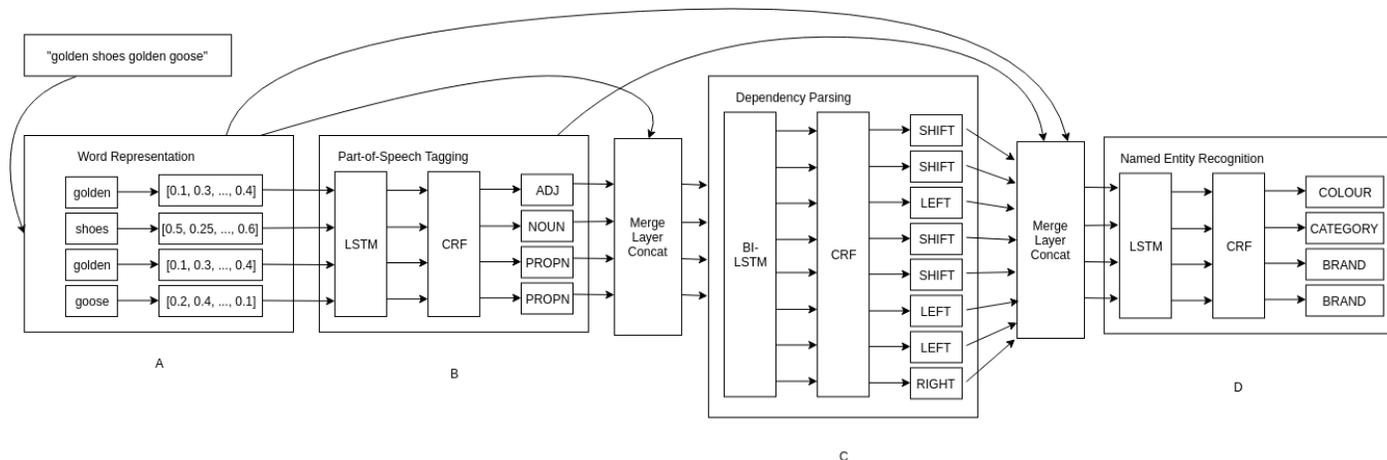


Figure 2: Semantic Search architecture diagram for the sentence “golden shoes golden goose”.

This module is an end-to-end sequence labelling solution, based on a Long-Short-Term-Memory (LSTM) combined with a Conditional Random Fields (CRF) classifier inspired in Xuezhe Ma and Eduard Hovy [12] work. It is trained from scratch using an internal dataset, generated from Farfetch products’ long descriptions. We argue that, in our platform, search queries are in fact a simplification of products’ long descriptions. By learning how to correctly classify long descriptions we are in fact a step closer to correctly classify each query. Figure 2 section [B] illustrates the module architecture. It takes as input a list of tokens embedded vectors. The next layer is an LSTM with hyperbolic tangent function (tanh) as activation. The recurrent activation is a hard sigmoid. The following layer is a CRF which returns the marginal probabilities on each time step. We use a RMSprop optimizer, an adaptive learning rate method proposed by Geoffrey Hinton [10]. The results and parameters details can be seen on Section 4.

### 3.3 Dependency Parsing

A syntactic analysis requires not only the identification of a Part-of-Speech tag for each word but also their relation. In this context, a Dependency Parser is responsible for creating a graph connection between each word.

We started by implementing a transition-based dependency parser, inspired by Chen and Manning [2]. Traditionally, this is a sequential task, in which the input is unfolded into a stack, a buffer and a list of operations. In the initial state, the buffer contains the list of embeddings of the sentence, while the other structures start empty. The stack works as an auxiliary structure, while the list of operations store all the predicted operations so far. During its run, an LSTM unit receives a state representative vector and generates the next operation in line. The next state is computed from the previous one in conjunction with current output. The final state is achieved when both the stack and the buffer get empty. Once reached, we can retrieve the final list of operations.

Although, we were able to mount a solution in a transitional manner, we felt the necessity of implementing a different model, due to the inherent speed problems, as described in Section 4.2.

Thus, inspired by Dyer’s [7] Stack-LSTM work, we have modified the architecture into an end-to-end sequence labelling problem. Contrary to our previous approach, where the model is run N times until the final state is reached, this new method only requires the recurrent network to run once. For a list of tokens we sequentially predict operations which get appended to the corresponding final list. We explored different architectures and found out a model composed by a Bidirectional-LSTM layer connected to a CRF layer provides more solid results. We also achieve a more robust outcome when using the Part-of-Speech tagging module output as feature. We apply a RMSprop optimizer learning rate method. Figure 2 section [C] illustrates the module architecture. The results and architecture details can be seen on Section 4.

### 3.4 Named Entity Recognition

At last, the Named Entity Recognition module seeks to detect and classify tokens into meaningful tags to Farfetch’s domain (e.g., Brand, Colour, Category). Its contributions are key in order to enrich our text interpretation capability. This module tightly connects our Natural Language Parser to the fashion domain. In terms of technical approach, the task is similar to Part-of-Speech tagging, where each token associates to its respective Named Entity tag. Our end-to-end sequence labelling solution, which explores a similar solution to Xuezhe Ma and Eduard Hovy [12] work, uses the previously described tasks outcome as features to increase the information flow into itself.

This module is an end-to-end sequence labelling LSTM-CRF, in which each input token vector representation is enriched with its Part-of-Speech tag and Dependency Parsing connections. In practice, we concatenate each token vector with an embedding representation of Part-of-Speech and Dependency Parsing features, token-wise.

Figure 2 section [D] illustrates the module architecture. A more detailed explanation and benchmark results are provided on the following Section 4.

Brand	Category	Short Description	Long Description
Y / Project	Jeans	High Waisted Jeans with Chaps	“Black cotton High Waisted Jeans with Chaps from Y/PROJECT featuring a button and zip fly, belt loops, a five pocket design, a straight leg and frayed edges.”
Givenchy	Bags	Studded Antigona Tote	“Dark red calf leather studded Antigona tote from Givenchy featuring a trapeze body, a gold-tone logo plaque, top handles, a gold-tone stud detailing, a main internal compartment, a full lining, an internal zipped pocket, an internal logo patch and a detachable and adjustable shoulder strap.”
Victoria Victoria Beckham	Shorts	Patterned Shorts	“Black cotton blend patterned shorts from Victoria Victoria Beckham.”
Fleur Du Mal	Lingerie	Charlotte lace denim thong	“Black and indigo blue cotton-blend Charlotte lace denim thong from Fleur du Mal. Underwear and lingerie must be tried on over your own garments.”

**Table 1: An example of product’s short and long descriptions.**

## 4 EXPERIMENTAL STUDY

### 4.1 Dataset

As noted throughout the paper, fashion domain only recently has received significant attention. In this regard, and to the best of our knowledge, the available data on the web for model benchmarking in this domain is scarce. Due to this, our analysis was restricted to Farfetch data. This data comes mainly from three main sources. One of the sources is products’ long and short descriptions. Upon each new product arrival, data collecting and generation process starts. In association with a novel product, we have short and long descriptions, consisting of a couple of paragraphs that textually describe the product’s characteristics - see Table 1. Another source is related with our user’s interaction with our search engine. We parse clickstream logs that for each query register its potential association with a clicked product. The last source is related to data augmentation techniques, which leverage existing product characteristics to create artificial content. Those characteristics range from brands to categories (different levels, e.g., *clothing* > *dresses* > *day dresses*), passing through attributes (such as colour) - see Table 1. Afterwards, with data generation techniques associated with template generation we are able to extend our useful vocabulary. We render over 91k unique words and 21.3M annotated sentences.

### 4.2 Experimental Setting

The Natural Language Parser architecture is depicted in Figure 2. Each one of the four modules on the architecture is trained separately, in a sequential order. After each module training phase, we freeze its weights and append the next module. We maintain a 9:1 train/test split across all experiments.

#### Word Representation:

Firstly, in our embedding layer, we use dense vectors techniques to achieve a semantic word representation. As pre-processing strategy, sentences were split into tokens along white-spaces. Our tokenizer scheme did not take more complex approaches into account, such as stemming or lemmatization, since that would remove important usable information from each sentence.

The creation of embeddings is a crucial step in our solution. They produce a robust word representation which serves as foundation for each one of the following modules. We only use short and long

product descriptions, where each word can be represented by its context.

We have made an extensive study with two well-known embedding techniques: word2vec[15] and GloVe[17]. The hyper-parameters fine tuning was set according to the original papers. We keep an embedding dimension of 300 across all models.

#### PoS:

In the Part-of-Speech tagging task, we aim to learn the correct syntactic annotation for each token. We argue that, in our platform, search queries are a simplification of product long descriptions. By correctly learning annotated long descriptions, the model should easily generalize the Part-of-Speech tags for our search queries. As training data, we used a pre-annotated dataset of our products’ long descriptions. State-of-the-art models, trained with formal English, should be able to generalize their results to our fashion context due to the non-domain specificity of the grammar. Standard models, however, usually rely on language specificities, such as capitalization, for getting higher output confidences. In our use-case, queries are introduced by users, who often neither write perfect sentences nor follow all the referred particularities. Thus, we annotate long descriptions with a state-of-the-art PoS tagging model and apply text normalization on top, so our input data is more representative of what exists in the real world. The objective is to capture the nuances of fashion language constraints in a more limited representation, which do not occur in classical datasets used to benchmark PoS tagging tasks [14].

Our PoS tagging solution (LSTM-CRF as presented in [12]) was trained with 2048 elements per batch, 512 steps per epoch and 32 validation steps. We use dropout layers for regularization. The LSTM layers has 100 units and the CRF is able to classify a total of 20 classes of Part-of-Speech tags, a common simplification from the 36 used in the Penn Treebank Project[14].

#### DP:

The Dependency parsing module takes Part-of-Speech tags and embeddings as features in order to learn the relationship between words, namely how they modify each other. Similarly to the PoS tagging module, we use the long descriptions corpus and their corresponding annotations. Yet again, it is necessary to train a tailor-made solution from scratch. As stated in Section 3, we’ve implemented two different modules to access this task. We discarded

our initial sequential solution, the transition-based iterative module, due to its low training speed. Although the problem is stated and addressed by [3], our data volume severely aggravates this limitation. As previously mentioned, the chosen model consists of an end-to-end architecture using a Bidirectional-LSTM to predict the full list of operations in a single iteration, in a many-to-many fashion. Comparing to the Part-of-Speech tagging neural network model, besides using the Bidirectional variant of LSTM, we also double the number of units to 200. The CRF layer is able to predict a total of four classes: the three types of operations present in a transition-based solution plus the unknown tag. During training we use 2048 elements per batch, 512 steps per epoch and 32 validation steps.

**NER:**

Finally, the Named Entity Recognition module is responsible for annotating a range of meaningful domain-specific tags, such as *Brands* and *Categories*. The data produced by each one of the previous models gets concatenated and used as input to the Entity Recognizer, taking full use of the hierarchical construction. Named Entity Recognition presented some serious challenges, mainly due to the lack of annotated data. Typical solutions do not work in our domain, so the obvious approach is to extrapolate tags from our long descriptions dataset. In order to increase the quality of results, we perform data augmentation via artificial generation as mentioned in Section 4.1. This way we could reduce length bias and the propagation of the unknown tag across the results. The best model consists of an LSTM of 100 units combined with a CRF layer, similar to [12] and to our Part-of-Speech tagging implementation. We consider 5 different tags of entities, namely *Brand*, *Category*, *Colour*, *Attribute* and *Unknown*.

**4.3 Results**

Task	Model	Features	Accuracy	F1
POS	LSTM+CRF	WORD	0.9820	0.9821
DP	BI-LSTM+CRF	WORD+POS	0.8079	0.8043
NER	LSTM+CRF	WORD+POS+DP	0.9902	0.9903
NER	LSTM+CRF	WORD+POS	0.9898	0.9898
NER	LSTM+CRF	WORD	0.9891	0.9891

**Table 2: Experimental Results for all modules in our hierarchical deep learning Natural Language Parser**

Table 2 presents a list of benchmarked models, and their characteristics, to be evaluated against Accuracy and F1 score metrics.

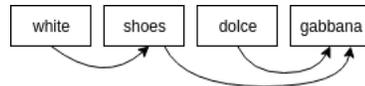
On Table 3 we can look at some examples of neighbouring words to common fashion vocabulary in our embedding space. We confirm related words seem to get mapped closer to each other. For example, in the word2vec solution, the word "hat" is closer to hat types, such as "beanie", "fedora" and "beret". The same happens using the brand name "Gucci", which strongly relates to other brands, such as "Fendi" and "Kenzo", in our vectorial space. Due to the unsupervised nature of these tasks, we must perform an extrinsic evaluation oriented to measurable tasks. In our case, we've studied the impact on our models, which behaved similarly for both word2vec and GloVe.

	word2vec	GloVe
'skirt'	('dress', 0.68)	('been', 0.86)
	('skort', 0.56)	('brass', 0.84)
	('shorts', 0.55)	('short', 0.83)
	('miniskirt', 0.55)	('that', 0.81)
	('culottes', 0.53)	('details', 0.81)
'hat'	('beanie', 0.73)	('loafers', 0.95)
	('fedora', 0.68)	('thanks', 0.91)
	('beret', 0.67)	('hinge', 0.91)
	('trilby', 0.61)	('thing', 0.90)
	('headband', 0.59)	('evokes', 0.90)
'Gucci'	('fendi', 0.69)	('fly', 0.97)
	('dsquared2', 0.64)	('la', 0.89)
	('moschino', 0.64)	('offers', 0.77)
	('kenzo', 0.63)	('the', 0.75)
	('givenchy', 0.62)	('le', 0.73)

**Table 3: An example of embeddings trained with Farfetch's data.**

We've decided to follow with word2vec in our Word Representation module.

In more detail, Table 2 presents the results for the Part-of-Speech tagging task. Alongside the standard LSTM model presented, we also tested its Bidirectional variant. We opted for the simpler architecture due to the lack of value provided by its extra parameters. Table 4 displays the results of ambiguous queries to our use case. One kind of challenge our system may encounter is the usage of 'golden', either to identify a colour or a brand. This is where the Part-of-Speech tagging may help reducing ambiguity, since these two forms can only be distinguish syntactically, either as an adjective or proper noun respectively. A similar problem can be seen for the word 'red' (colour or the *Red Valentino* brand).



**Figure 3: Dependency Parsing results for the sentence "white shoes dolce gabbana".**



**Figure 4: Dependency Parsing results for the sentence "red dress from red valentino".**

Again, Table 2 describes the results associated with our best model for the Dependency Parsing task. Contrary to what was registered in the Part-of-Speech tagging solution, the Bidirectional formulation played a crucial role in this task. In Figure 3 and 4 we can see some specific results of our model.

Finally, in Table 2 we present the Named Entity Recognition results. While training we noticed that our models were not able

query	result				
'golden shoes golden goose'	<b>golden</b> (‘ADJ’, 0.85)	<b>shoes</b> (‘NOUN’, 0.94)	<b>golden</b> (‘PROPN’, 0.80)	<b>goose</b> (‘PROPN’, 0.95)	
'red dress from red valentino'	<b>red</b> (‘ADJ’, 0.99)	<b>dress</b> (‘NOUN’, 1.00)	<b>from</b> (‘ADP’, 1.00)	<b>red</b> (‘PROPN’, 1.00)	<b>valentino</b> (‘PROPN’, 1.00)

Table 4: Part-of-Speech tagging.

query	result			
'golden shoes golden goose'	<b>golden</b> (‘COLOUR’, ‘0.99’)	<b>shoes</b> (‘CATEGORY’, ‘0.66’)	<b>golden</b> (‘BRAND’, ‘0.88’)	<b>goose</b> (‘BRAND’, ‘1.00’)
'black gucci bag'	<b>black</b> (‘COLOUR’, 0.63)	<b>gucci</b> (‘BRAND’, 1.00)	<b>bag</b> (‘CATEGORY’, 0.64)	

Table 5: Named Entity Recognition.

to deal with the dominant ‘unknown’ tag present in our sparse long descriptions annotations. The template generation was key to increase the results quality. We achieved better results with a complete architecture using all the information as feature to our model, connected to the LSTM-CRF model. Although the results are extremely similar, those slight variations are the difference between solving ambiguities in a correct way. Table 5 present some examples queries to that model.

## 5 CONCLUSION

Despite more than two decades of research and development on search engines, solutions tailored for the fashion domain are still feeble. Interpretation of textual queries written by users cannot be approached by a keyword-based solution alone. We provide a solution to comprehend the domain with effective identification of fashion entities.

In this article we have described a hierarchical deep learning Natural Language Parser that takes advantage of transfer learning techniques and specialist models to learn fashion vocabulary and be able to segment and recognize fashion entities in the wild. Although we focused on the fashion-domain, this approach can easily be generalized to other contexts.

Our empirical results establish a robust baseline, which justifies the use of hierarchical architectures of deep learning models. Farfetch now has the ability to interpret text in a novel way enabling the search engine to build much more precise and non-ambiguous queries. Ultimately, this leads to a much better user experience while opening new paths of research to provide tailor-made results.

## ACKNOWLEDGMENTS

The authors would like to thank all team members of the search department from Farfetch. Thanks to their support, we could assess the quality of our algorithms and the usability advantages provided to our customers. The first author would also like acknowledge Ernesto Costa from Universidade de Coimbra for supporting his work while at Farfetch.

## REFERENCES

- [1] Thorsten Brants. 2000. TnT: a statistical part-of-speech tagger. *Proceedings of the sixth conference on Applied Natural Language Processing* (2000), 224–231. <https://doi.org/10.3115/974147.974178>
- [2] Danqi Chen and Christopher Manning. 2014. A Fast and Accurate Dependency Parser using Neural Networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 740–750. <https://doi.org/10.3115/v1/D14-1082> arXiv:1512.00567
- [3] Qian Chen, Xiaodan Zhu, Zhenhua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2016. Enhanced LSTM for Natural Language Inference. 2008 (2016), 1657–1668. <https://doi.org/10.18653/v1/P17-1152> arXiv:1609.06038
- [4] Jason P C Chiu and Eric Nichols. 2015. Named Entity Recognition with Bidirectional LSTM-CNNs. (2015). arXiv:1511.08308 <https://arxiv.org/pdf/1511.08308.pdf><http://arxiv.org/abs/1511.08308>
- [5] Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing. *Proceedings of the 25th international conference on Machine learning - ICMML '08* (2008), 160–167. <https://doi.org/10.1145/1390156.1390177> arXiv:1603.06111
- [6] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural Language Processing (Almost) from Scratch. *Journal of Machine Learning Research* 12 (2011), 2493–2537. <https://doi.org/10.1.231.4614> arXiv:1103.0398
- [7] Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. 2015. Transition-Based Dependency Parsing with Stack Long Short-Term Memory. (may 2015). <https://doi.org/10.3115/v1/P15-1033> arXiv:1505.08075
- [8] Yoav Goldberg. 2016. A Primer on Neural Network Models for Natural Language Processing. *Journal of Artificial Intelligence Research* 57 (2016), 345–420. <https://www.jair.org/media/4992/live-4992-9623-jair.pdf>
- [9] Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2016. A Joint Many-Task Model: Growing a Neural Network for Multiple NLP Tasks. (2016), 1923–1933. arXiv:1611.01587 <http://arxiv.org/abs/1611.01587>
- [10] Geoffrey E. Hinton, Nitish Srivastava, and Kevin Swersky. 2012. Lecture 6a-overview of mini-batch gradient descent. *COURSERA: Neural Networks for Machine Learning* (2012), 31. [http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture6\\_slides\\_lec6.pdf](http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture6_slides_lec6.pdf)
- [11] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of Tricks for Efficient Text Classification. In *ACL*, Vol. 2. 427–431. <https://doi.org/1511.09249v1> arXiv:1607.01759
- [12] Xuezhe Ma and Eduard Hovy. 2016. End-to-end Sequence Labeling via Bidirectional LSTM-CNNs-CRF. (mar 2016). <https://doi.org/10.18653/v1/P16-1101> arXiv:1603.01354
- [13] Christopher D. Manning, Prabhakar. Raghavan, and Hin Schütze. 2009. *An Introduction to Information Retrieval*. Cambridge University Press. 482 pages.
- [14] Mitchell P Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics* 19, 2 (1993), 313–330. <https://doi.org/10.1162/coli.2010.36.1.36100>
- [15] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. (2013), 1–12. <https://doi.org/10.1162/153244303322533223> arXiv:1301.3781

- [16] Wenzhe Pei, Tao Ge, and Baobao Chang. 2015. An Effective Neural Network Model for Graph-based Dependency Parsing. 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)*. 313–322. <https://doi.org/10.3115/v1/P15-1031>
- [17] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. GloVe : Global Vectors for Word Representation. (2014), 1532–1543.
- [18] Stephen Robertson. 2010. The Probabilistic Relevance Framework: BM25 and Beyond. *Foundations and Trends® in Information Retrieval* 3, 4 (2010), 333–389. <https://doi.org/10.1561/1500000019>