Package and Classify Wireless Product Features to Their Sales Items and Categories Automatically

Haitao Tang, Pauliina Eratuuli

Package and Classify Wireless Product Features to Their Sales Items and Categories Automatically

Haitao Tang and Pauliina Eratuuli

Commercial Management and Business Digitalization, Nokia, Finland
haitao.tang@nokia.com

Abstract. Aiming at automated decision making, this paper defines and analyzes two machine learning use cases for the product process in wireless infrastructure business. The first use case assigns a product to a product packet according to the functionality of the product. The second use case determines the category of the product so that it can be priced. Then, the product is ready for sale. This paper also provides solutions to these machine learning use cases. The solutions are examined with real data from the processes. The credibility of the solutions is also evaluated by comparing the machine learning decisions with the decisions of human users. These human users know the actual assignment and classification of those products. The results show that the solutions work well as they expected. These solutions assign and classify a part of the given products fully automatically with a high confidence and accuracy. Due to insufficient prediction confidences for the rest of the given products, the rest part of products needs to be escalated for the further decision by the human users. With an escalation, a set of assignment and classification options for a given product is also recommended by the solutions. Often, the correct assignment and classification exist in the set of options already. The human users can easily identify and select the correct assignment and classification from the recommended options. Significant costs and processing time can thus be prevented.

Keywords: Natural Language Processing, NLP, Machine Learning, ML, Process Automation, ML based decision making, LTE, 5G, Business Digitalization, Pricing.

1 Introduction

Providing cellular communication products is the major business of a telecommunication infrastructure vendor. The products include cellular network products of Long-Term Evolution (LTE) [1] and 5th Generation (5G) [2], which can be in the forms of Hardware (HW), Software (SW), or their supporting components. The products are made available for sale through the process of product packaging, classifying, and price setting

The internal reference price (IRP) setting is an internal product process that is conducted to define all needed pricing related attributes for such a product before it is

released for sale. This is currently a manual process which is repeated for hundreds of products annually. During the IRP setting, the category classification and sales package of the product need to be made correctly. In many cases, a new product should be assigned to an existing sales package that contains similar products.

If not automated, this process involves heavily human evaluation and decision making. In such a manual process, a human user needs to understand the whole product landscape completely, which includes not only the various available products and the products expected to be coming, but also the detailed functionalities of the products, their relations, and their relevance to the different network service operators. This process is not only time consuming, but also requires a high level of experience and knowledge from the human user. The good side-effect of such a manual process is that human experience and knowledge are also encoded and embedded into the data generated during the process. During the years of manual processing, it has created the critical amount of data. These data could be used by machine learning to release human from such tedious and brain-straining manual process.

Any commercial digitalization project should be based on a business need. After identification of a possible use case, the business case should be validated. For the automation of the IRP setting, the business need is not only to reduce the time spent on the price setting process, but also to increase the quality of the process to a high level regardless of the user's expertise level. The motivation of this work is thus to design the Machine Learning (ML) solutions to automate the IRP setting process. It defines the ML-based IRP setting process that can dramatically reduce the time and competences required to set the IRP prices. It also increases the quality of the process to a high level regardless of the competence level of the human user.

The ML-based solutions are achieved by using Natural Language Processing (NLP) and general-purpose ML methods to assist the decision making for the product classification and sales package assignment. ML is used to identify the closest existing matching package and category for a given product. The matching is done based on the description documents of the products.

This paper is organized as the following. A brief review of ML-based NLP is given in Sec. 2. The use cases of this work are defined in Sec. 3. The actual method to assign a product to its corresponding package is presented in Sec. 4. The actual method to classify the product category is depicted in Sec. 5. Sec. 6 presents the experiment setting and results. The credibility of the trained models is further analyzed in Sec. 7. The conclusions of the work are given in Sec. 8.

2 Statistical Natural Language Processing

NLP [9, 10] is a multi-discipline field supported by computer science, linguistics, and machine learning technologies. It concerns the ML-based learning, understanding,

extraction, representation, and producing of data in human languages. NLP has greatly benefited from the recent advances in machine learning. It is now focusing on how computer can do speech recognition, natural language understanding, and natural language generation.

Speech recognition translates human speech into text. Natural language understanding interprets and extracts the text of human languages. Natural language generation produces text and speech in human languages. The typical NLP methods could be categorized as text preprocessing, semantic vectorization and embedding, Neural Network (NN)-based parsing of text and information extraction, as well as deep-learning based encoding and decoding of representations of a set of texts.

The methods of text preprocessing perform object standardization, text tokenization, stop-word removing, token (e.g., word) stemming, and token lemmatization. The methods of semantic vectorization and embedding mapping can map a set of texts to their corresponding vectors based on token frequency in one form or another. It can also model topics through latent analysis of a set of texts. It can embed the words in a set of texts, as well as embed a set of texts as bags of words and word sequences. The NN-based parsing methods parse text into parse trees of the sentences including their part of speech tagging. Then, the methods of information extraction extract named entities from text, relations between named entities, and knowledge from text. The methods of deep-learning based encoding and decoding of representations use either mainly RNN-based sequence to sequence models or attention-based transformers to encode and decode representations of texts. It appears that the attention-based transformers outperform RNN-based models in general purpose and multi-task applications.

With the NLP methods, numerous NLP applications can be realized, e.g., the methods in [10, 11]. They are, for example, applications of sentiment analysis, question answering, language modeling, detecting semantic textural similarity, language generation, document summarization, and machine translation.

3 Definition of the Use Cases

IRP setting is done for the products of the different telecommunication technologies. Each technology needs its own ML models to be trained with the technology specific data, as the sales structures for different technologies differ significantly.

There are two use cases in the decision automation. Full automation of the IRP setting process is for the products to which the prediction confidence and accuracy levels exceed predefined thresholds. In the case of ML assisted decision making, the information on the products will be presented to a human user, together with their ML based proposals for the categories and sales packages. Then, the user makes the final decision with the help of ML prediction and assignment. These two use cases can be combined. Full automation can be made for those product cases with the high prediction

confidence and accuracy levels. For those product cases with low confidence levels, ML recommends the category and sales package for human's final decision.

The data of this work are the documents defining the products as well as the available sales packages and SW categories. The documents are written by human for the purpose of product implementation and product sales. Typically, a corpus of the product documents is collected per technology family, which usually has thousands of the documents. The corresponding sales packages and SW categories of the products are the ground truth data. They have been generated during the process of sales item creation during the past years. There are hundreds of such labeled data points available per technology family. It is worth to mention that the documents are written in a peculiar, technology-specific language. They are full of "special" technical terms and abbreviations, as well as local conventions. This makes it not possible to directly use a pre-trained language model of the general purpose (e.g., spacy [7] and BERT [8]). Specific language model must then be trained for this work.

This work applies the NLP solutions to complete two tasks. Task A embeds the documents of the human generated product descriptions into their corresponding vectors semantically. The embedding enables the detection of the functional similarity between two products. Such detection is necessary to properly assign a product to a sales package according to its functionality.

Task B classifies a product to a proper category according to the description of the product. It applies a classifier, which is built at the end of the ML pipeline. Text descriptions of the products are input to the pipeline. Basic NLP preprocessing of the texts is then made and, the texts are mapped into numerical vectors as input to the classifier. The SW categories of the products are used as the target output. The classifier is then trained accordingly. Finally, the trained model is used to predict the SW category of a given new product.

4 Method to Package a Feature

The packaging of a product is realized as what is shown in Fig. 1. After the preprocessing of the product feature documents, we use the NLP document-to-vector solution [13] to embed each of the already packaged products (if any) in a sales item list. The actual embedding model is trained with all the available product documents. This embedding model can then represent the products well. It makes the similarity comparison between two products in the IRP list more accurate than what could be achieved with a general-purpose embedding model. Please note that the texts (documents) in the IRP list are only a subset of the texts of the product documents.

The reason for using the similarity-based assignment instead of a categorical classifier is that it needs to assign among several hundred packages. A classification-based

method usually achieves a rather low accuracy when there are only hundreds of data samples available.

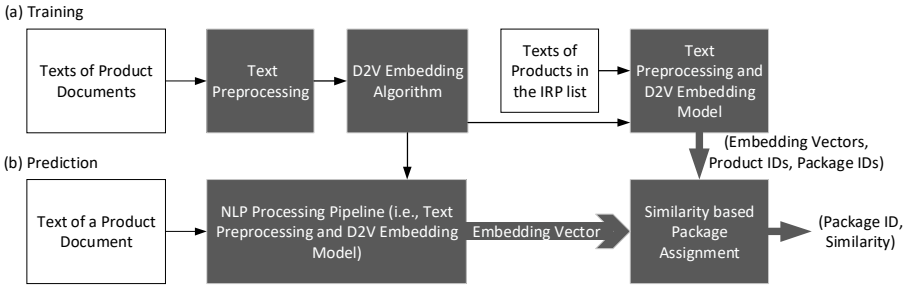

Fig. 1. Assign a product to a sales package according to the semantic similarity.

The exemplary results of the generated embedding and assigned packages are presented in Table 1. The products unambiguously similar to each other are assigned to the same package. Otherwise, the products are each assigned to an individual one-product package. When a new product arrives for the package assignment, the embedding vector of the new product will be compared with all the vectors of the existing products in the table. If there is an unambiguously similar product existing in the table, the new product is assigned to its package. Otherwise, the new product is assigned as an individual product into a new package.

Table 1. The example embedding information of products.

Package ID	Product ID	Embedding Vector
0	D_x	*(0.78, -1.50, -0.6, -0.19, -0.11, 0.52, 1.13, 0.77, -0.36, 0.22, -0.19, -0.39, 0.26, -1.83, 0.84, -0.66, 0.73, 0.37, 1.05, -0.43)*
…	…	…

Whether to make fully automated assignment or not depends on the required accuracy of the above ML-based assignment. If the above assignment provides an accuracy higher than the requirement, the assignment is done fully automatically. Otherwise, the ML-based assignment serves as a recommendation for the human decision maker. It is up to the human to decide the actual assignment based on the assignment recommendation. The details concerning these options are introduced in Section 6.

5 Method to Assign a SW Category to a Product

The classification of a product to its SW category is realized as shown in Fig. 2. After the preprocessing, a TF-IDF (Term Frequency–Inverse Document Frequency) vectorizer is trained with the corpus of all the available documents of products. Now, the trained TF-IDF model has the vocabulary of all the available documents. Another TF-IDF vectorizer is created by using this vocabulary. The second TF-IDF vectorizer fits

and transforms the processed texts of the products in the IRP list into numerical vectors, one for each product in the IRP. The products in the IRP list are the already packaged products (if any) in a sales item list. Together with their known SW categories, their vectors are used to train the multi-class classifier.

When there is a request to classify the SW category of a new product, the preprocessing of the document of this product is made first. The processed text of the document is then fed to the second TF-IDF vectorizer, which fits and transforms the processed texts of the given product into its corresponding numerical vector. This vector is then fed to the trained multi-class classifier. The classifier predicts a SW category for the vector (i.e., the new product) with a specific confidence (i.e., prediction probability). The reason to use a categorical classifier here is that it can achieve a good accuracy for the classification among a small number of classes, when there are hundreds of data samples available.

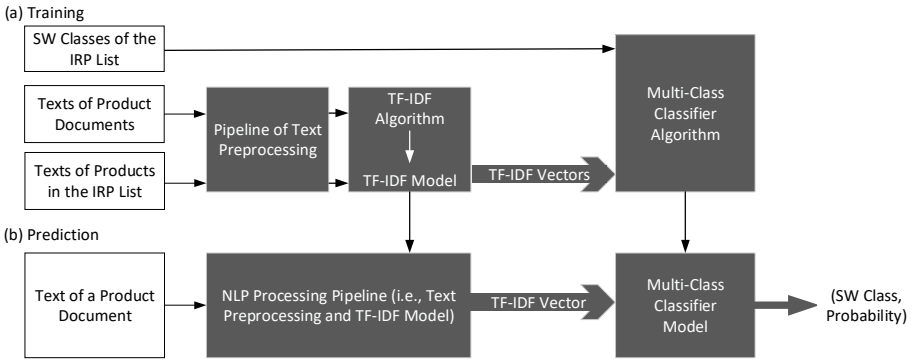

Fig. 2. Assign a product to a SW category according to its functionality and importance.

It is worth mentioning that the method shown in Fig. 2 is suitable for the cases where enough data samples (even if not huge) are available for training the multi-class classifier. However, in an extreme case, the number of product documents for specific category or categories can be very small. In this situation of data scarcity, a categorical classifier may not work. It is simply because of the lack of enough training data for the categorical classifier. For example, as shown in Fig. 3, there are very few Class II product documents to train a multi-class classifier properly. In such a case, the methods shown in Fig. 1 and Fig. 2 could be used together as an ensemble method. The ensemble method could still bring an acceptable "classification" result. There could be extra information in the embedding model as it is trained with a bigger corpus of all available products. The extra information could thus improve the accuracy via an ensemble method.

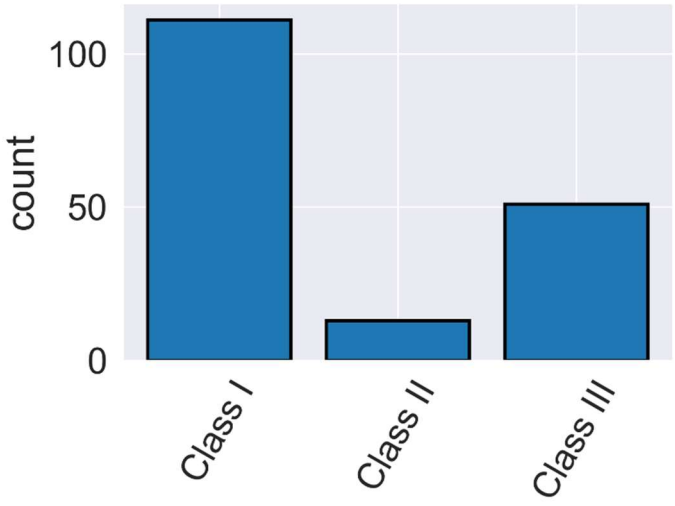

Fig. 3. The extreme example counts of certain available product documents for the SW classes, Class I (63%), Class II (8%), and Class III (29%).

6 Experiments and Results

The solutions based on the methods described in Sec. 4 and Sec. 5 are realized with Python 3.6 and its corresponding ML libraries. The solutions are trained with real product data and then they predict the sales item package and the SW category for a given new product. The data, experiments, and results of the solutions are presented and discussed in this section.

6.1 Introduction of the Real Data

The first part of the real data for the solutions are the product documents for telecommunications technologies. For each technology, there are thousands of such documents, which are written in telecom-technical English by R&D people in the company. Such documents could each have the length from a few paragraphs up to multiple pages. Their combined vocabulary of words/terms are at the level of ten thousand. Often, parts of the technical context are not directly given in the documents. A reader is assumed to know the technical context (domain knowledge) before the reader could fully understand the semantic content of the documents. This assumption adds challenge to the solution when comparing with general-purpose NLP tasks [7, 8], where the huge amount of available data could compensate the missing context information. In addition, full scale object standardization for the documents is not feasible due to e.g. the existence of inconsistent abbreviations and varying technical terms.

The second part of the real data for the solutions are the sales items packages in the IRP lists. They provide the information of the package IDs and the SW categories of the

products in the IRP lists. This part of information annotates the first part of the data. The products in the IRP lists are just a part of all the available product proposals.

6.2 Package Assignment for a Product

The text preprocessing in Fig. 1 is realized through (1) removing stop-words and punctuations and (2) partial object standardization. The Doc2Vector (D2V) embedding algorithm [13] utilized in this application is from the gensim library. It is trained with all the available product documents, where the dimensions (20 and 24) of the embedding space bring the best performance for the examined technologies correspondingly.

There are 243 different sales packages in a real IRP list used as the existing package to test the solutions. This IRP list has about 500 different products. The distribution of the products among these packages is shown in Fig. 4. Each of the packages has only a small proportion (maximal 5.74%) of the total products.

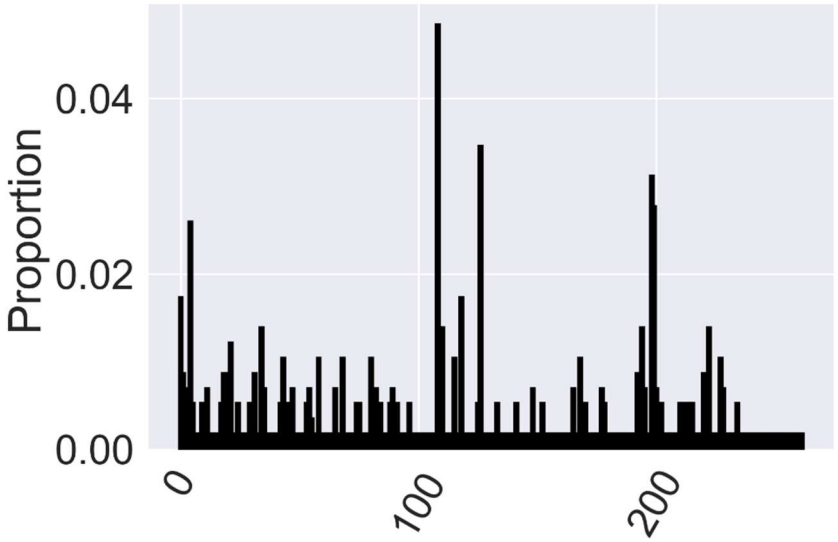

Fig. 4. The distribution of products among the packages in an existing IRP list, where the x-axis is the package ID and the y-axis is the proportion of products in a package to the total number of products in the IRP list.

A sequence of 42 new products are then assigned one by one, by the solution shown in Fig. 1. 28 of them are assigned to existing packages and 14 of them are assigned as new packages. Whenever a new product is assigned, it is added to the existing IRP list. The existing IRP list is extended with the newly assigned product. The next new product will be assigned according to the extended IRP list.

The experiment shows that 25 (60%) of the new products are assigned correctly to either the existing packages or as new packages themselves. When recommending a new

product to a package, the solution also provides the top 0 to 5 existing products (if any) that are the most similar products to the given new product (i.e., the top matching products in the existing IRP list). For 35 (83%) of the new products, the correct package information is among the provided top similar products from the existing IRP list.

Usually, one cannot trust the solution to assign the new products fully automatically as there are only 60% of the products assigned correctly by the solution. Human in the loop is thus required in this case. The human needs to review a recommendation from the solution and decide the package for the new product. However, the work for the human is very much easier now when comparing with the work when a human alone makes an assignment. In the pure human assignment, the human user needs to know and remember all the products and their packages in the existing IRP list. The human assignment work takes a lot of the time to search and check against the products and packages in the existing IRP list. When using the ML solution as recommendation, the human user can immediately identify the correct package information from the top matching products provided for the newly given product by the solution. Then, the human user can simply select the correct package from the top matching products. In this way alone, 83% of the new products can be assigned correctly. For the remaining part of new products, the human user still has to search through and check against the products, product documentation and packages in the existing IRP list.

The D2V embedding model needs to be retrained after every *n* new product documents have been released by R&D. These newly generated product documents can carry extra information that has not been learned by the former embedding model. The *n* can be any large number as long as the newly generated product documents do not contain any new product to be assigned to a sales package. This means the former trained embedding model has still enough information for the new product to be assigned. Otherwise, the D2V embedding model needs to be retrained before the actual assignment of the new product. As the R&D process is not very fast, it is usually enough to retrain the D2V embedding model once every week. In case that a product feature progresses from its creation to the sales item assignment in less than a week, the D2V model is retrained on demand.

6.3 Assignment of SW Category for a Feature

There are 4 different SW categories in a real IRP list. The IRP list has about 500 different products. The distribution of the products among these SW categories are shown in Fig. 5. If one predicts the SW category for a newly given product always with the SW category having the largest number of products, the prediction accuracy could be about 36.1%. It is low.

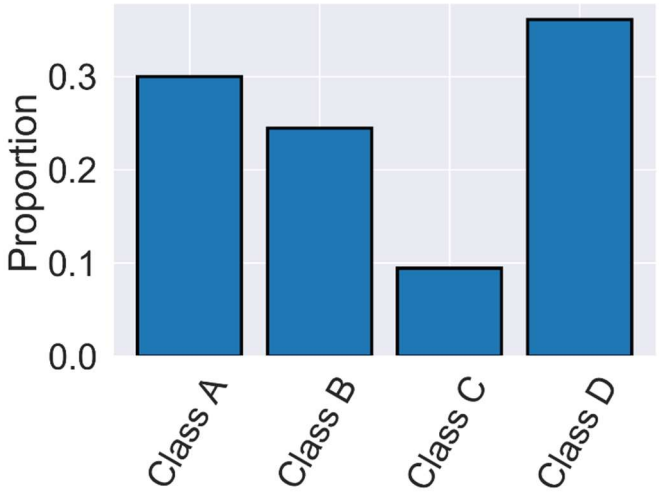

Fig. 5. The distribution of the products among the SW categories.

The text preprocessing in Fig. 2 is realized through (1) removing stop-words and punctuations and (2) partial object standardization. The TF-IDF algorithm is first fitted with all the available thousands of product documents. The vocabulary of this trained model is used by the TF-IDF algorithm as the vocabulary when fitting with all the product documents of the existing IRP list. The TF-IDF model also generates the TF-IDF vectors for all those product documents. These vectors together with the SW categories of those products are used to train a multi-class logistic regression algorithm. The trained model is then used to predict the SW category of a newly given product.

When there is a newly given product for SW category prediction, the text of the product document is preprocessed. Then, the TF-IDF model transforms the preprocessed text into its corresponding TF-IDF vector. This vector is then input to the multi-class classifier model. The model thus predicts the SW category of the newly given product. The model also provides the prediction probability of the predicted SW category.

499 product documents of an IRP list are put through the "(a)" training process of Fig. 2, which eventually trains the multi-class classifier. The trained multi-class classifier model is used to predict the SW categories of another 125 products. The accuracy to predict the SW categories of these 125 products is 82.4%. The other prediction scores for these 125 products are given in Table 2.

Table 2. The prediction scores except the accuracy score for the 125 products.

	Precision	**Recall**	**F1-Score**	**Support**
Class A	0.80	0.95	0.87	38
Class B	0.86	0.58	0.69	31
Class C	0.90	0.82	0.86	11

Class D	0.82	0.89	0.85	45
Micro Avg	0.82	0.82	0.82	125
Macro Avg	0.84	0.81	0.82	125
Weighted Avg	0.83	0.82	0.82	125

The normalized confusion matrix of the predictions is shown in Fig. 6. This multi-class classifier did not predict Class B very well. Here, 13 Class B products in its total 31 products are wrongly classified to Class A (7) and Class D (6).

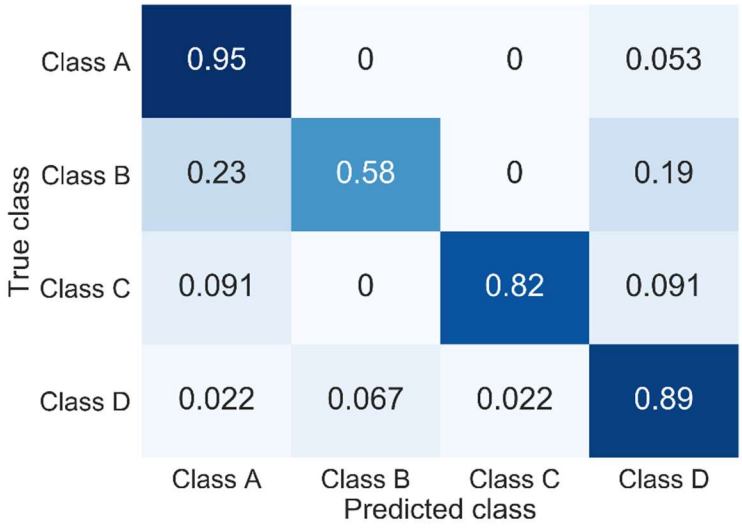

Fig. 6. The normalized confusion matrix on the prediction of the 125 products.

The classification results against their corresponding prediction probabilities by the multi-class classifier are summarized in Fig. 7. A correct classification has indeed a clear correlation with a high prediction probability. However, it is hard to differentiate the correct and incorrect predictions simply by checking the prediction probabilities when the probability score is lower.

It is thus possible to enable the fully automatic classification of the products with high prediction probability. To do so, one can provide a confidence threshold on the prediction probabilities. When a prediction probability is larger than the threshold, the classification can be considered as acceptable and fully automatic classification is triggered. When a prediction probability is smaller than the threshold, the classification can be considered as not trustable and the case is escalated for human to evaluate and classify. As shown in Fig. 8, assume the confidence threshold is set to be prediction probability 0.65. In this case, 91 (72.8%) of the 125 products can be automatically classified. The classification accuracy of this part of products is 91.2% (i.e., 83 products). The confidence threshold is set according to the specific business needs. It is usually selected with a given classification accuracy value that are minimally acceptable to the business.

The selection is a tradeoff between the fully automatic classification and machine learning assisted classification.

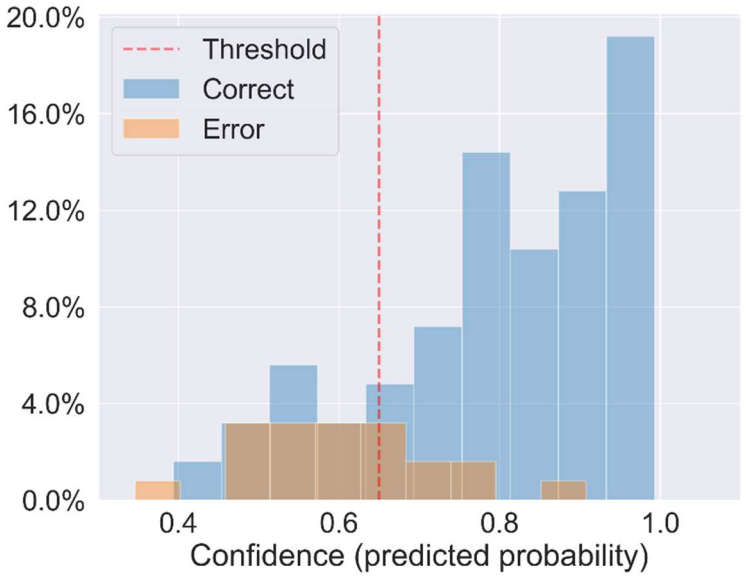

Fig. 7. The distributions of the correct and incorrect classifications of the 125 products against their prediction scores.

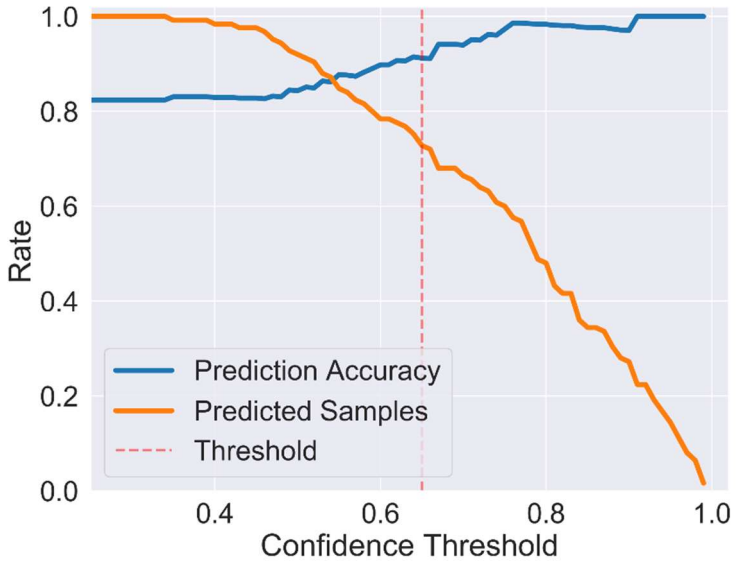

Fig. 8. The tradeoff between the classification accuracy and the actual number of products classified automatically, where confidence threshold determines the tradeoff.

A cross validation on the quality of the multi-classifier is made with 100 times of re-shuffling the combined 624 products, 80% for training and 20% for testing. The mean classification accuracy is 0.763 and the standard deviation is 0.03. The accuracy distribution of the cross validation is shown in Fig. 9.

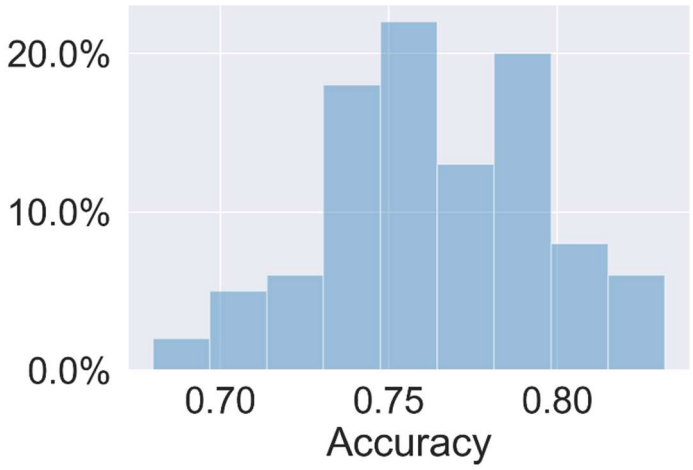

Fig. 9. The accuracy distribution of the cross validation.

The multi-class classifier needs to be retrained after every n new products have been classified and assigned to the IRP lists. These newly classified products can carry extra information that has not been learned by the previously trained model. If the n is large, the classification accuracy could suffer clearly. If the n is too small, the re-training can be too frequent. Depending on how frequently a new product needs classification, the higher the retraining frequency, the higher the n value. For the experiments made above, it would be good to let $n = 10 \sim 15$.

The vocabulary of the TF-IDF model needs to be re-fitted only when the newly generated product documents by R&D contain any new product to be classified. This means the formerly fitted TF-IDF does not have enough vocabulary information for the new product to be classified. As the R&D process is not very fast, it is usually enough to re-fit the TF-IDF vocabulary once every week.

7 Credibility of the Trained Models

The quality and credibility of an embedding model can be evaluated with a set of benchmark product documents, each with a similar product document scored by human beforehand. This evaluation method uses the query inventory method [15], while the query here is not on a word but on the text of a product document. For example, Table 3 shows one query point (from the set) with the human scored similarity and, the model-inferred similarities when comparing the vector of document D_k to its inferred vector

and the inferred vector of D_x. In this example, we could conclude that the model infers well for this query point. It is thus a good model for document D_k and D_x. More query points can be evaluated to assure the quality of the trained model. It is also mostly doing well for other query points in the benchmark product documents. We could conclude the trained embedding model is good.

Table 3. The quality of the embedding model for a given product document D_k when compared with the document D_x.

Similarity scored by human	*('D_k',1)*	*('D_x', 0.9)*
Similarity inferred by trained model	*('D_k', 0.974)*	*('D_x', 0.923)*

One also needs to know if the trained multi-class classifier has made the classification with the proper information in the product documents, and not with something irrelevant. The model is trustable if the evaluation confirms that. This evaluation is made with the lime library [12]. As shown in Fig. 10 and Fig. 11, the classifier (TF-IDF and the multi-class classifier) uses the relevant texts when it classifies a product. In Fig. 10 and Fig. 11, the probabilities for the SW categories (named as Class A, Class B, Class C, and Class D) are predicted. The contributing terms and text sections are also shown to support or oppose predictions of the SW categories, together with their numeric levels of the contribution.

The lime-based evaluation of the 125 products have been made with the same approach as shown in Fig. 10 and Fig. 11. The evaluation needs the domain knowledge concerning the relevance of the information and what information indicates a specific SW category or opposes it. The evaluation results on the 125 products show that the relevant information in the texts have been correctly used to predict the categories of these products in most cases. The classifier of this solution is thus considered trustable. For example, terms “allocation”, “prb”, and “block” have contributed correctly to support the classification of Class D.

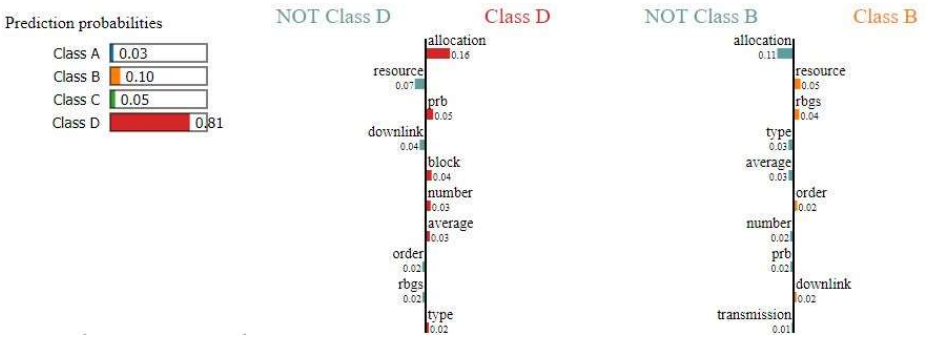

Fig. 10. The prediction probabilities of the four SW categories and the contribution terms for or against the three SW categories, Class B, Class C, and Class D, concerning an example product.

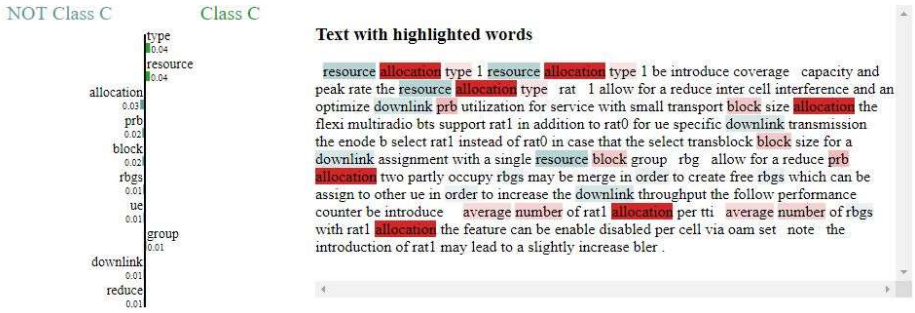

Fig. 11. The contributing terms for or against the SW category Class D and the text used by the classifier to classify the SW category of the example product described by the text.

8 Conclusions

This work has proposed the ML solutions to realize the automated IRP setting process. Experiments are taken to explore the feasibility and performance of the solutions, given the available data. The results show that the solutions work well as expected by the human users. They are enough to assist human in the decision making, which reduces significantly the processing time, the needed competence, and human-caused errors. Under a given prediction confidence threshold, these solutions can also fully automate the IRP setting for those products where their prediction confidences by the solutions are higher than the given threshold.

The credibility of the models is further evaluated against the texts of product descriptions and the human provided benchmark of similar products. The trained embedding model infers the top-two similar products mostly as given in the human benchmark. The text components used to predict the product categories are mostly those key elements in the documents of product description.

The ML solutions are provided as a web service to the whole IRP setting process. The request and response attributes of this Application Interface (API) are rather general. There is no need to change the interface even if there is an update for the ML solutions. This makes the ML solutions modular.

Through the experiments, it is also found that the prediction accuracy is generally increasing with the amount of available assignment data. The data volume increases with the usage of the solutions. The performance can be further improved with additional data.

The ML solutions are designed for the products per technology. For those technologies with rather limited amount of existing data, more complicated ML solutions would be needed to achieve the required performance. As new data comes daily, there will be a need to evolve the machine learning solutions at certain point of time. Extra data can

reduce the need of a complicated model in one hand. On the other hand, it can also enable the application of a more advanced model to achieve an even better performance. However, it needs further work and experiments to find the exactly needed balance when sufficient amount of extra data become available.

References

1. 3GPP: LTE, https://en.wikipedia.org/wiki/LTE_(telecommunication).
2. 3GPP: 5G, https://en.wikipedia.org/wiki/5G.
3. Kurama V.: Introduction to Machine Learning, Towards Data Science (https://towardsdata-science.com/introduction-to-machine-learning-db7c668822c4), Jul 2017.
4. Patal A.: Machine Learning Algorithm, Overview, Medium (https://medium.com/ml-research-lab/machine-learning-algorithm-overview-5816a2e6303), Jul 2018. ----
5. Ding S., Zhu Z., Zhang X.; An Overview on the Semi-Supervised Support Vector Machine, Neural Computing and Applications, Springer, Vol. 26, No. 8, Nov 2015. ----
6. Arulkumaran K., Deisenroth M. P., Brundage M., Bharath A., A.: A Brief Survey of Deep Reinforcemnet Learning, IEEE Signal.Processing Magazine, Special Issue on Deep Learning for Image Understanding, pp.1-16, Sept 2017. ----
7. spaCy: Industrial-Strength Natural Language Processing, https://spacy.io/ ----
8. Devlin J., Chang M. W., Lee, K., Toutanova, K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, arXiv:1810.04805v1, https://arxiv.org/pdf/1810.04805.pdf, Oct 2018. ----
9. Bird S., Klein E., and Loper E.: Natural Language Processing with Python, NLTK (https://www.nltk.org/book/), Jul 2015.
10. Davydova, O.: 10 Applications of Artificial Neural Networks in Natural Language Processing, Medium (https://medium.com/@datamonsters/artificial-neural-networks-in-natural-language-processing-bcf62aa9151a), Aug 2017.
11. Brownlee, J.: 7 Applications of Deep Learning for Natural Language Processing, machinelearningmastery (https://machinelearningmastery.com/applications-of-deep-learning-for-natural-language-processing/), Sept 2017.
12. Ribeiro M. T., Singh S., Guestrin C.: "Why Should I Trust You?" Explaining the Predictions of Any Classier, arXiv:1602.04938v3 [cs.LG] (https://arxiv.org/pdf/1602.04938.pdf), Cornell University, Aug 2016.
13. Le, Q., Mikolov, T,: Distributed Representations of Sentences and Documents, Proceedings of the 31st International Conference on Machine Learning (ICML), pp. 1188–1196, 2014.
14. Lau J., Baldwin, T.: Practical Insights into Document Embedding Generation, Proceedings of the 1st Workshop on Representation Learning for NLP, pp.78-86, Aug 2016.
15. Schnabel T., Labutov, I., Mimno, D, Joachims T.: Evaluation Methods for Unsupervised Word Embeddings, Proceedings of Conference on Empirical Methods in Natural Language Processing, pp. 298–307, Sept 2015.
16. Frey, B.J., Dueck, D.: Clustering by Passing Messages Between Data Points, Science, Vol 135, pp.972-976, Feb 2007.
17. Rousseeuw, P.J.: Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis, Journal of Computational and Applied Mathematics 20 (1987) 53-65.
18. Madakam, S., Holmukhe, R. M., Jaiswal, D. K.: The Future Digital Work Force: Robotic Process Automation (RPA), Journal of Information Systems and Technology Management, Vol. 16, 2019.