Cross-Media Knowledge Acquisition: A Case Study

Marina Giordanino[1], Claudio Giuliano[2], Damjan Kužnar[3], Alberto Lavelli[2], Martin Možina[3], and Lorenza Romano[2]

[1] CRF, Torino, Italy,
marina.giordanino@crf.it
[2] FBK-irst, Povo TN, Italy,
giuliano|lavelli|romano@fbk.eu
[3] University of Ljubljana, Slovenia,
martin.mozina|damjan.kuznar@fri.uni-lj.si

Abstract. The paper describes an approach to cross-media knowledge acquisition which combines text and raw data. The approach has been applied in a real-world use case concerning wind tunnel reports within the EU-funded project X-Media. The goal is to identify the source of wind noise in a vehicle and find the most suitable solution to reduce it. Information is extracted from the textual parts of the reports and provided to the raw data tool to improve its performance. The results of the initial experiments are encouraging.

1 Introduction

Human perception relies on visual, raw data, and textual stimuli but also on the exploitation of domain specific background knowledge. This is true also in industrial domains where engineers report experimental results in technical reports that exploits different media in an integrated way. The inherent specificity and concision of this kind of technical documents induce the experts to fully use multimedia contents. They create contents that require the synergy of different human perceptive functions in order to be fully perceived and thus, the combination of different communication sources (i.e., images, texts, raw data) strictly interlinked. As a consequence, the intended meaning can be effectively transferred only considering such sources in a combined/fused manner. One of such examples concerns the identification of the source of wind noise in a vehicle, and the proposal of the most suitable technical solution to reduce noise in terms of aerodynamic noise comfort. In this context, the analysis of past test results may allow experts to know how to design suitable solutions for new products avoiding new long and costly experiments. In particular, these results allow wind tunnel experts to know which vehicle components are crucial for reducing noise (e.g. mirrors, windshield, etc.).

The data and expert knowledge were provided by Fiat Aerodynamic department in the context of the EU X-Media project. Competitors' vehicles undergo several tests to measure their performance, while during their development Fiat

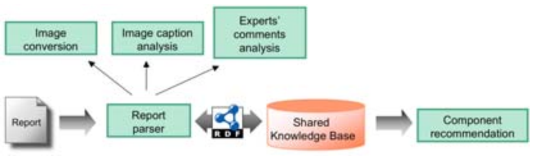

Fig. 1. Overall schema of the modules involved in the use case.

vehicles are tested to verify if they achieve the required target for each performance. The process is long and costly as they need to test each possible component to determine its influence on the interior noise. In this paper, we describe a method that estimates noise produced by a component prior to testing. One of the main challenges is related to the high complexity of the task, due to a number of factors: First, dozens of competitors' and Fiat vehicles tested and technical reports produced by wind tunnel experts every year. Technical reports are semi-structured Word documents containing graphs and tables with expert comments. Second, different kinds of aerodynamic measurements taken: for each vehicle, for each of eight microphones placed inside the vehicle, for each component contributing to noise, and in different stages of the product development process. Third, extraction, analysis and classification of vehicle-related information on: vehicles features and configuration, test results from noise spectra, tables, and their parameters, components involved and quality of contribution to noise. Finally, estimation of components' influence on noise for new vehicles. The last element is particularly challenging. It requires the prediction of a full noise spectrum, which consists of 110 continuous values. Moreover, as already mentioned, experts provide comments of previous tests' results, saying whether component's influence on noise is high or negligible. We shall investigate whether such information can be useful (or how useful it is) for the predictions of noise curves.

An overall schema of the modules involved is shown in Figure 1. From the report images (2d graphs), the noise curves and captions are extracted (image conversion). These are thus processed by the image caption analysis that, analysing the caption of the image, identifies the type of experiment reported in the curve. In the last step of textual analysis, the tool finds sentences containing experts' comments about vehicle components and classifies them as *critical* (component increases noise) or *non-critical* (component reduces noise). Finally, the data extracted from the document and saved on the knowledge base are used to build a model that can predict influence of a component in a new (not yet tested) vehicle. The entire process is ontology-based. A Fiat ontology has been designed describing the whole set of concepts involved and their properties and relations e.g. vehicle, component, wind tunnel test etc. The first four steps related to extraction of data from documents are described in Section 2, while we present the method for prediction of a noise spectrum in Section 3. We finish the paper with conclusions and directions for further work.

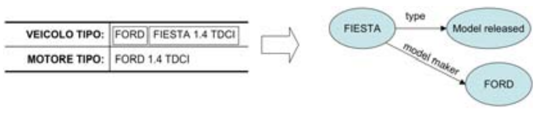

Fig. 2. Extraction of vehicle characteristics from tables.

2 Text Extraction

After performing the wind tunnel tests, experts produce reports consisting of: (a) textual tables containing basic vehicle characteristics: vehicle model, maker, segment, size, etc; (b) experts' summaries of experiment results: a set of paragraphs (text) where experts express their opinions about components used in the vehicle: which of them turned up to be good with respect to noise reduction and which of them should be improved; (c) experiment results: a set of noise curves/spectra and tables containing noise levels in the vehicle in different conditions.

Experts' opinions reported in summaries contain relevant information and could, if used together with methods for knowledge acquisition (KA) from raw data, improve the prediction accuracy of single media models. Moreover, curve captions and tables contain textual information necessary to the raw data tools to associate their output with the concepts present in the ontology.

We identified the following three interactions between the text and raw data tools: (1) The tools for text and raw data extraction need to map their output into the ontology. To this end, the textual KA tools extract the vehicle model and maker from the report (Figure 2). This task is performed using simple regular expressions and, since the information is contained in tables, the process is almost 100% correct. (2) Another key information is the mapping between the spectra and the ontology. This is achieved performing entity recognition on the graph captions (Figure 2) and associating the spectra with the corresponding experiments. As training we use a set of entities presents in the ontology, using the k-nearest neighbor algorithm where the Levenshtein distance is used to compute the distance between two entities. The obtained accuracy is around 85%. This information is also used to enrich the feature space of raw data tools and, as we will see in Section 3.3, contributes to improve the acquisition process. (3) The information extracted in this last step is exclusively used to improve the acquisition process (Figure 4). The information extracted from the report summaries is used as background knowledge by the raw data tools. The summaries contain the salient aspects of a vehicle test, in particular, the vehicle components that have been tested and their negative or positive influence on the vehicle noise (i.e., if the components are critical or not). The k-nearest neighbor algorithm is also used to extract the car's components, while the noise influence is estimated using a strategy similar to sentiment analysis. We used a small set of manually-labeled summaries to train the classifier. The overall accuracy is around 51%.

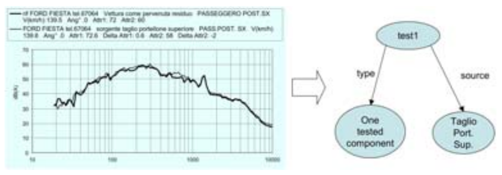

Fig. 3. Caption classification. The output is used for ontology mapping by the modules performing cross-media KA.

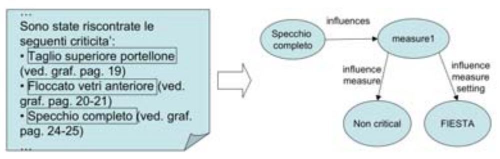
...
Sono state riscontrate le seguenti criticita':
• Taglio superiore portellone (ved. graf. pag. 19)
• Floccato vetri anteriore (ved. graf. pag. 20-21)
• Specchio completo (ved. graf. pag. 24-25)
...

Fig. 4. Component recognition and classification. The output is used as background knowledge in the modules performing cross-media KA.

Note that this result is strongly influenced by the accuracy of the component recognition (75%).

3 Noise prediction

3.1 Problem definition

A single learning example used in noise prediction consists of: (a) a vehicle shape wind noise spectrum, shortly *vs_spectrum* (here all component-related noise is removed by fully taping critical parts of a car), (b) an original vehicle noise spectrum, shortly *o_spectrum* (noise measured in the original car), (c) name of a tested component, (e.g., front door cut) (d) a vehicle wind noise spectrum with one component being tested, shortly *comp_spectrum*, and (e) expert's comments on the *comp_spectrum* (possible values are *critical* and *non-critical*).

The "raw" type of data are the three noise spectra (a,b,d), which are automatically extracted from graphs in the Image conversion tool (e.g., Figure 5). As these images are in vector graphics format, there is no need to use "true" image analysis tools, but we can parse images' source code with a few regular expressions to extract the noise spectra. Each noise spectrum is thus represented with 110 measurements, where one measurement presents the sound pressure level (in dB) for each octave in $0-10^4$ frequency range. The name of the tested component is extracted from the image caption, as described in the previous section. The expert comments are extracted for each component from the description written in natural language classified into *critical* and *non-critical* values. A component classified as critical is a component that has a large impact on wind noise. In this paper, we shall use manually classified comments, since the results of automatic

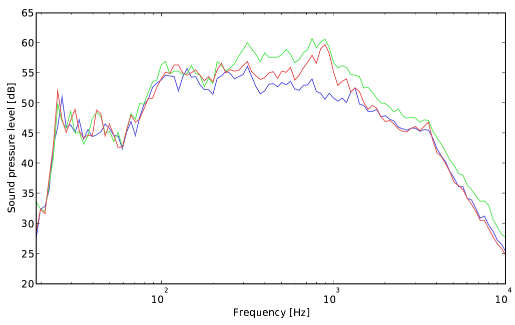

Fig. 5. Comparison of noise spectra in original vehicle (green), vehicle shape (blue) and vehicle with one component being tested (red). On vertical axis is sound pressure level (measured in dB), on horizontal is frequency (Hz).

extraction are currently not reliable enough (51%), and cannot be successfully used in learning. The current data set contains 186 learning examples parsed from 18 documents (=18 tested cars; approx. 10 experiments per car).

The task is to estimate the level of noise change for each component proposed by an expert, when this component is untaped; in other words, we need to predict the *comp_spectrum*. We will measure two different types of prediction errors. The first is root mean squared error (RMSE) between the predicted and the true noise curve, and the second is the relative (by car) number of inconsistencies with respect to expert comments. An inconsistency can occur between predictions of two different components of the same vehicle, where one is critical and one non-critical, and predicted noise change is higher for the non-critical component. The change of noise is defined as Euclidean distance between the *comp_spectrum* of the component and the *vs_spectrum* of the tested car. To estimate these two errors, we used a leave-one-vehicle-out strategy (i.e., each time the examples associated to one vehicle were left out). The list of possible components for each vehicle tested was the same as those that were actually used in the experiments.

3.2 Methods using only raw data

In this section, we report on the results obtained from raw data only; we predict the *comp_spectrum* using *o_spectrum* and *vs_spectrum* only. Since raw data does not contain information about the actual tested component, we cannot expect to achieve good results, however, it is still possible to make a prediction based on average influences in the testing data. The results obtained here will be used as baseline for comparison with the cross-media methods.

The simplest methods used are **Shape** and **Original**. Shape is a method that always predicts the spectrum of the vehicle shape (*vs_spectrum*) and therefore assumes no influence of components on noise. Similarly, Original is a method that always predicts *o_spectrum*.

The next method used is **AverageInfluence**. This method first computes the average relative influence of all components in the learning set, and then applies it to each of the test examples. The relative influence of a single component (example) i is computed as

$$rel\_inf_i(freq) = \frac{comp\_spectrum_i(freq) - vs\_spectrum_i(freq)}{o\_spectrum_i(freq) - vs\_spectrum_i(freq)}$$

where $comp\_spectrum(freq)$, $o\_spectrum$, and $vs\_spectrum(freq)$ are sound pressure levels in dB at given frequency $freq$ for one tested component, original vehicle and vehicle shape, respectively. Average relative influence avg_rel_inf is merely the average over all relative influences for all components in learning set. The $comp\_spectrum_k$ of a new component k is then computed with the following formula:

$$\begin{aligned} comp\_spectrum_k(freq) = {} & vs\_spectrum_k(freq) + \\ & + diff_k(freq) \times avg\_rel\_inf(freq) \end{aligned}$$

where $diff_k(freq)$ stands for $o\_spectrum_k(freq) - vs\_spectrum_k(freq)$.

SimpleKNN method adopts the basic idea from classical KNN. We believe that KNN is the most reasonable approach in this domain, since we need to predict 110 highly dependent continuous values. A noise spectrum is a continuous curve, therefore the differences between predicted adjacent points must be small, and KNN (or any other instance-based method) will always satisfy this constraint. Distance between two examples is calculated as:

$$dist(e_i, e_j) = dist(o\_spectrum_i, o\_spectrum_j) + dist(diff_i, diff_j)$$

where $dist$ is the Euclidean distance function. We use the distance to calculate the weight of a learning example e_i in KNN prediction of a new testing example e_k: $weight(e_i) = 1/distance(e_i, e_k)$ [2]. The relative weighted influence of a new example e_k is then computed by:

$$w\_rel\_inf_k(freq) = \frac{\sum_{i=0}^{n} weight(e_i) \times rel\_inf_i(freq)}{\sum_{i=0}^{n} weight(e_i)},$$

where n equals the size of train dataset. Using the weighted relative influence we can then calculate the prediction of $comp\_spectrum$ of the new component:

$$\begin{aligned} comp\_spectrum_k(freq) = {} & vs\_spectrum_k(freq) + \\ & + diff_k \times w\_rel\_inf_k(freq) \end{aligned}$$

We can see that the prediction of SimpleKNN method is similar to the Average-Influence method. The difference is that SimpleKNN applies relative influences more selectively by using the measure of similarity between vehicles. This means that the similar vehicles (based on Euclidean distance) contribute more to the predicted component influence.

LearnDistanceKNN is an improvement of previously described SimpleKNN. It uses a generalised formula for distance

$$dist(e_i, e_j) = w_1 \times dist(o\_spectrum_i, o\_spectrum_j) + w_2 \times dist(diff_i, diff_j)$$

where w_1 and w_2 are weights that are optimised to achieve lowest prediction error (RMSE) on learn data. We used a simple hill-climbing based method for optimisation. Table 1 shows results of used methods. Shape and Original have the worst results, as expected. The best result is achieved by SimpleKNN method,

Table 1. Prediction errors of methods using only raw data.

Method	RMSE	Avg. inconsistencies
Shape	15.50	0.43
Original	28.52	0.27
AverageInfluence	10.97	0.28
SimpleKNN	10.73	0.22
LearnDistanceKNN	10.95	0.27

Table 2. Prediction errors of methods using raw data and component names.

Method	RMSE	Avg. inconsistencies
SimpleKNN(with comp.)	10.74	0.22
LearnDistanceKNN(with comp.)	9.49	0.17
LearnDistanceKNN(with c.&comments)	9.42	0.16

3.3 Cross-media approach

In the previous section we described methods using raw data only. Here, we extend them to accept the complete learning example that contains, along to the raw data, also the name of used component and experts' comments on experiment results. To use component names we simply added them to the feature space and extended SimpleKNN and LearnDistanceKNN methods with a third term (for component name) in the distance computation. The distance between two components is defined as 1, if they are different, and 0, if they are the same. The results are shown in Table 2. As you can see, extending SimpleKNN does not result in a better model, since the distances between components are not contributing enough. On the other hand, the new LearnDistanceKNN predicts significantly better. After inspecting weights learned for distance computation, we can see that the weight for component name is always higher (approx. 10-times) than weights for spectra differences.

Using expert comments in learning is more difficult. Expert comments distinguish components in two classes: critical and non-critical. Critical components are components that do not reduce wind noise well enough with respect to the vehicle shape wind noise. While it is important to have as accurate models as possible, it is also important to distinguish between critical and non-critical components in the same vehicle, since critical components are the weak points in the vehicle that have to be improved.

The criticality cannot be used as a feature in learning, because it is not available for new vehicles (expert can not state its criticality until they conduct experiments). Therefore, we need to use this feature as a form of background knowledge in the process of learning. We applied the principle used in QFilter [3]. QFilter is a method used in qualitative reasoning that makes quantitative predictions, which are guaranteed to be consistent with the induced qualitative model. These predictions are often considerably more accurate than those obtained by the state-of-the-art numerical learning methods. QFilter changes class values of examples to make them consistent with a given qualitative model. In our case, the class value is relative influence, which we decrease if the value is too high for non-critical examples, and increase low values of critical examples. In this paper, we arbitrarily used 0.1 as a threshold; therefore, whenever a critical component has relative influence at any frequency less than 0.1, its value is increased to 0.1, and when a non-critical component has relative influence at given frequency higher than 0.1, its influence is decreased to 0.1. This changes class values in a way that makes the difference between critical and non-critical examples bigger.

We applied the mentioned change to the LearnDistanceKNN (with component name) method. The RMSE of the method improved to 9.42, while the average relative inconsistency decreased to 0.16. As expected, using text comments decreased the number of inconsistencies. At the same time the accuracy of the method did not decrease, which is a very good result, although surprising. One would expect that changing class values in learn data would result in a decrease of total accuracy.

4 Conclusions and Future Work

In this paper, we have described the initial attempts to improve raw data analysis exploiting information extracted from text in a use case involving reports describing wind tunnel experiments. We have shown that, by using text extracted comments, we improve the average rate of inconsistency, therefore there are less mistakes in ordering the components by their influence on noise. Moreover, the results also show an improvement in RMSE, which is promising. In general, there is still a large space for improvement in raw data extraction, text extraction only, and in their combination. One possibility is to use more training data. In the current evaluation we have used only 18 reports. We expect to have about 100 parsed by the end of the project and analyzed. We also plan to improve our algorithms by considering similarity of different components (e.g. two different components can have a similar effect on noise spectrum) computed from their influences on the training data.

Acknowledgements

This work has been funded by the X-Media project (www.x-media-project.org) sponsored by the European Commission as part of the Information Society Technologies (IST) programme under EC grant number IST-FP6-026978.

References

1. M. Giordanino and G. Scarafiotti. Deliverable D13.1. Testbed description and requirement analysis for competitor analysis. Deliverable, X-Media Consortium, 2006.
2. D. Shepard. A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings of the 1968 23rd ACM national conference*, pages 517–524, 1968.
3. D. Šuc, D. Vladušič, and I. Bratko. Qualitatively faithful quantitative prediction. *Artificial Intelligence*, 158(2):189–214, 204.