TROMPA: Towards Richer Online Music Public-domain Archives

Deliverable 4.1

Crowd Evaluation Methodologies

Grant Agreement nr	770376
Project runtime	May 2018 - April 2021
Document Reference	TR-D4.1-Crowd Evaluation Methodologies v1
Work Package	WP4 - Crowd Annotation and Incentivisation
Deliverable Type	Report
Dissemination Level	PU - Public
Document due date	30 April 2020
Date of submission	28 May 2020
Leader	TUD
Contact Person	Julián Urbano (j.urbano@tudelft.nl)
Authors	Alessandro Bozzon (TUD), Geert-Jan Houben (TUD), Jaehun Kim (TUD), Cynthia Liem (TUD), Christoph Lofi (TUD), Ioannis Petros Samiotis (TUD), Julián Urbano (TUD)
Reviewers	Lorenzo Porcaro (UPF), Juan Sebastián Gómez (UPF)

Executive Summary

The aim of Task 4.1 – “Crowd-powered Improvement” is to design and implement a framework for the continuous evaluation and improvement of the automatic technologies in WP3, as well as their combination with the crowd to enrich music content. This task builds on top of the workflow definitions in Task 4.4 – “Campaign Design” and the user modeling in Task 4.2 – “Annotators”. This document presents a formal framework and demonstrates its application in the context of Music Emotion Recognition systems developed in Task 3.2 – “Music Description”.

Large-scale and sustainable data enrichment is only achievable through the use of automatic technology capable of producing new data descriptions. In a traditional setting, this is achieved in four general phases: annotation of a training corpus; development of systems with this corpus; evaluation of system performance and possible refinement; and enrichment of a target corpus through the application of these systems. However, if a new kind of descriptors is to be used and new technology created for it, or if existing technology has to be adapted to classical music, ground truth data is needed to train systems and evaluate their quality. Crowdsourcing may be a viable alternative to collection annotations, but in the long term, and in an evolving and large-scale setting like TROMPA, sustainability can be at risk. It is therefore very important that the crowd provide annotations where they are needed the most. The setting laid out for TROMPA consists in: annotating a few examples of the target corpus to be enriched; develop or refine systems with these data; train a combination model to estimate pending annotations based on the output from systems, known annotations and other external sources of data; use these estimated annotations to enrich the corpus and estimate system performance; and determine which examples will be annotated next such that they maximize value in a new iteration in terms of system development, enrichment, or evaluation precision.

We exemplify this framework through the example of a K-class classification problem. Traditional system performance metrics like Precision, Recall and F-score require a full set of ground truth annotations, so they have to be adapted to missing data. In particular, we model uncertainty in the annotations through random variables, from which expected value and variance may be computed. The uncertainty in the annotations propagates to the metrics, which are now also determined by random variables. Therefore, system performance may be estimated via point and interval estimates, provided that we are capable of estimating annotations somehow. Having the crowd provide annotations has a twofold effect: the number of instances pending annotation is smaller and therefore human enrichment coverage is higher, and more training data is available to train the combination model. Annotation estimates from the combination model will be more accurate and precise, and so are estimates of system performance.

To determine which music items should be annotated first, we define multiple criteria depending on what is to be optimized. If the objective is to improve system performance evaluation, we decide to give priority to music items that are expected to have a large effect on system scores. If the objective is to provide systems with better training data, we give priority to items showing biased system behavior according to their confusion matrices. If the objective is to improve data enrichment, we give priority to music items for which the combination model has low confidence.

For the development of the combination model we consider a multi-tier structure to exploit different kinds of features. We envision multiple feature sets categorized in two dimensions: interval vs. external, depending on whether they consider only data from the task or external to it (eg. metadata); and static vs. dynamic, depending on whether they exploit known annotations or not.

Richer set of features should be able to make better predictions, which is particularly important for features of identifying nature such as some metadata, because they sometimes lead to very strong relationships that systems would not be able to find by themselves. The implementation of each of the models at different tiers may be achieved through a variety of classification algorithms, provided that they make probabilistic predictions. In this deliverable we consider uniform and empirical distributions as non-informative and informative baselines, respectively, as well as multinomial regression, classification trees, random forests, and support vector machines.

To illustrate the TROMPA evaluation methodology, we present an experiment for the task of emotion recognition, which links directly to the Music Enthusiasts use case. We follow the traditional dimensional approach for describing emotion in four valence-arousal quadrants. We use three datasets and four different neural systems developed in the context of Deliverable D3.2 – "Music Description". With respect to the external features to be used by the combination model, we compute a total of twelve general and high-level descriptors from the audio.

In a first experiment, we evaluate the performance of the combination model as a function of the number of available annotations. Overall, these results suggest that a combination model implemented with random forests will perform best, while a model implemented with multinomial regression will probably be overconfident and bias the evaluation results. A model implemented with support vector machines is likely to perform well once some annotations are available.

In a second experiment we study the behavior of the whole framework for evaluating systems in the face of missing annotations. To start the process, we assume the campaign initiator is able to provide two instances from each class to fit a first combination model, and from that point on the framework enriches the corpus using the predictions from the combination model, estimates system performance and establishes a priority for the next items to be annotated. The results show that estimation error rapidly approaches zero error with a few dozen annotations and largely stays within a 5% error margin. The variance in the predictions nicely decreases as more annotations are made, and except when there are very few annotations, they largely cover the estimation error as expected.

Using the case of emotion recognition as an example, we show how the framework can successfully estimate system performance even when there are just a few dozen annotations, unbalanced datasets, and biased systems. Additionally, we show that the final enrichment produced by this hybrid framework achieves twofold performance compared to a simple ensemble approach over the raw algorithm output.

Version Log		
#	Date	Description
v0.1	20 May 2020	Initial version submitted for internal review
v0.2	26 May 2020	Revised version after internal review
v1.0	28 May 2020	Final version submitted to EU

Table of Contents

1. Introduction

1.1 Scope

The aim of Task 4.1 – “Crowd-powered Improvement” is to design and implement a framework for the continuous evaluation and improvement of the automatic technologies in WP3, as well as their combination with the crowd to enrich music content. This task builds on top of the workflow definitions in Task 4.4 – “Campaign Design” to model the evaluation and enrichment processes under the uncertainty caused by missing ground truth data, and how this uncertainty propagates to the estimation of system performance. Building on top of Task 4.2 – “Annotators”, different prioritization criteria are also defined to determine which music items would be most beneficial if annotated, and their incorporation in a continuous evaluation workflow which improves efficiency and enables scalable datasets. The present deliverable presents a formal framework and demonstrates its application in the context of Music Emotion Recognition systems developed in Task 3.2 – “Music Description”.

1.2 Outline

The document is organized as follows. Section 2 explains and motivates the need for a hybrid crowd-machine framework and outlines the approach adopted in TROMPA. Section 3 presents the TROMPA System Evaluation framework by first describing how traditional evaluation is carried out, and how it is generalized to settings with missing or no annotations. Section 4 then elaborates on how to appropriately establish priorities for further annotation, such that the data provided by crowds has maximum impact. Section 5 describes the implementation of the combination model and its features. Section 6 presents results on several experiments to evaluate the presented framework, and Section 7 concludes the deliverable.

2. System Evaluation and Data Enrichment

Large-scale and sustainable data enrichment is only achievable through the use of automatic technology capable of producing new data descriptions. In a traditional setting, this is achieved in four general phases, as depicted in Figure 2.1:

1. **Annotation.** Some training corpus x_r is available to build systems, and a group of experts is dedicated to the creation of annotations a_r pertaining to some kind of descriptors for said data. These datasets are usually available for typical tasks so that they may be reused in various applications.
2. **Development.** Using the available training dataset, a set of systems s is trained following some sort of Machine Learning algorithms and different sets of parameterizations. Given new data, these systems are now able to enrich them by making predictions of the kind they were trained with.
3. **Evaluation.** These systems are of course subject to error, and so a natural question to ask is how well they perform when making predictions. Based on the annotations a_r and the system predictions y_r on the training dataset, some metric scores m are computed to assess

system performance. These evaluation results can then be used to determine how to further improve the systems and hence the enrichment.

4. **Enrichment.** The best performing system is then chosen, or a good-performing set of systems is assembled into a single system c, which is then applied to the target corpus x_t to produce the enrichment through their predictions y_t.

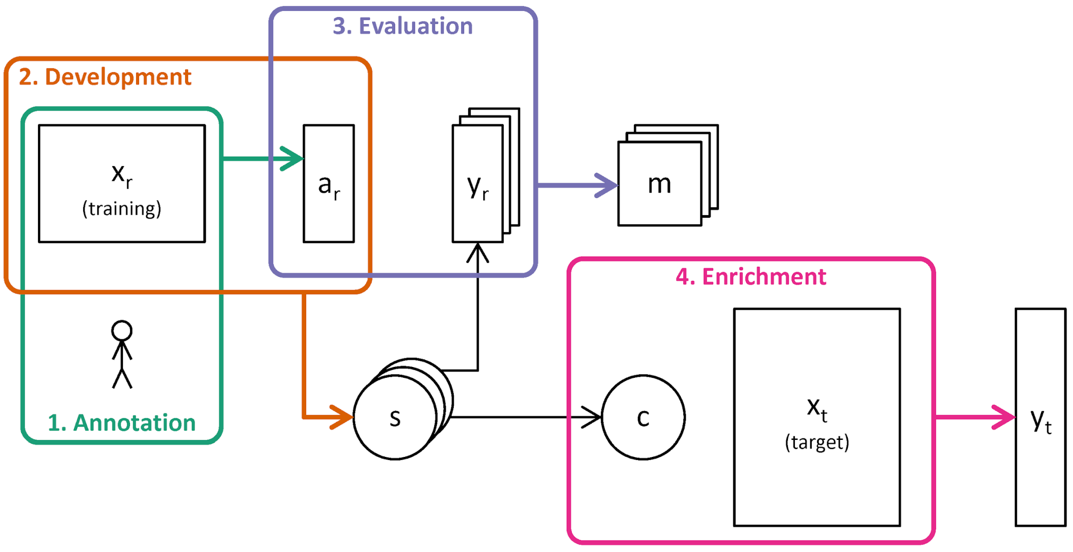

Figure 2.1. Traditional data enrichment at large-scale through the development of technology trained with existing data.

However, if a new kind of descriptors is to be used and new technology developed for it, ground truth data is needed to evaluate their quality. In a typical case, this technology will be based on Machine Learning methods, so ground truth data will likely be needed also for their very development through training datasets. Alternatively, an existing task may be such that appropriate systems do exist, but are specialized on a different type of data such as pop music. New ground truth data will be necessary to assess how well they work and, probably, to tune them to classical music. Gathering such data with domain experts may be prohibitively expensive or plainly impossible in the face of low resources. Resorting to crowdsourcing may alleviate the resource problem, but in the long term, and in an evolving and large-scale setting like TROMPA, sustainability would be at risk.

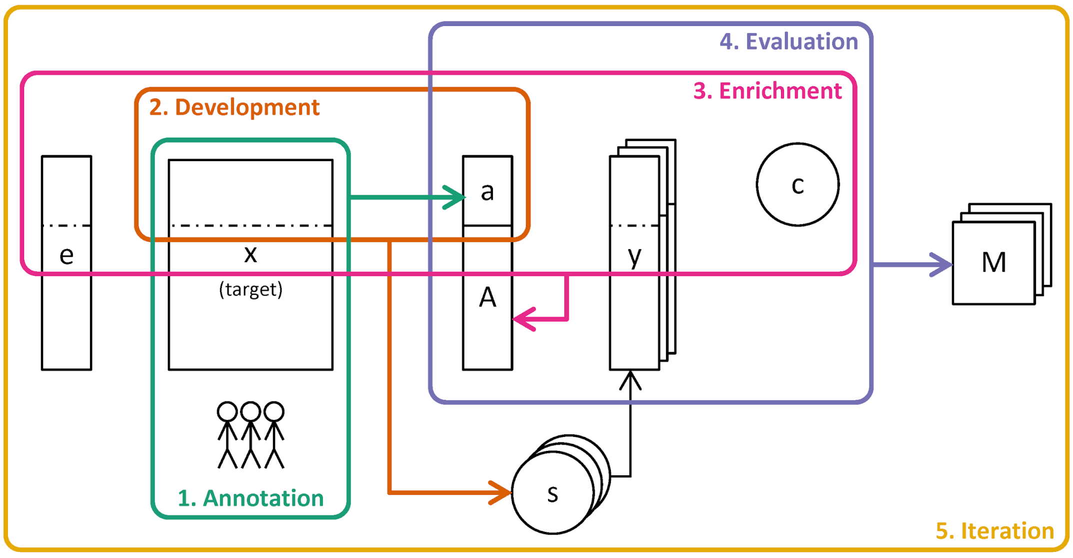

Figure 2.2. TROMPA data enrichment at large-scale through the continuous involvement of the crowd in the development of tailored technology trained without existing data.

It is therefore very important that the crowd provide annotations where they are needed the most, such as a piece of music where the automatic technology is likely to have failed, or where they have low confidence in their predictions. The setting laid out for TROMPA is depicted in Figure 2.2:

1. **Annotation.** Faced with the absence of training data, the group of experts and the crowd provide some annotations a only for a small fraction of the target corpus x. Most of the corpus remains unannotated and unenriched.
2. **Development.** The small set of annotations can still be used to bootstrap the development of systems or to tailor existing systems to the target corpus. Similarly, a set s of systems may be produced, whose quality is expected to be low given the small amount of data they are trained with.
3. **Enrichment.** The developed systems may still provide their predictions y for the whole target corpus. In addition, there may be other external sources e related to the corpus, such as metadata or other tasks or technology unrelated to the current enrichment space. Such data sources, along with the system predictions and known annotations, may be leveraged to build a combination model c to predict further annotations. Such a model encompases multiple sources of information about the corpus, and provides predictions A for the part of the corpus pending human annotation, thus enriching the corpus in a hybrid crowd-machine way.
4. **Evaluation.** Both system predictions and combination model predictions will be erroneous to some degree. In the case of systems, the known annotations a and predicted annotations A may be used to estimate system performance M, which will itself bear some degree of error. In the case of the combination model, performance may be estimated by comparing the known annotations a with the predictions it would have made for that part of the corpus. This can be used as an estimate for the overall enrichment performance.
5. **Iteration.** Unless the whole target corpus is enriched, the crowd may provide further annotations. As more annotations are incorporated (phase 1), the systems may be further improved (phase 2), which ultimately leads to an update of the combination model and the

corpus enrichment (phase 3), with similarly updated performance estimates (phase 4). Therefore, the whole setting will further improve as the crowd provides further annotations.

Therefore, the goal for involving the crowd in the TROMPA annotation activities is twofold:

1. The continuous evaluation and improvement of technologies developed within TROMPA.
2. The combination of humans and algorithms for multimodal description and enrichment of classical music data.

In order to achieve these goals, TROMPA envisions three annotation objectives:

1. **Evaluation.** Facing a lack of ground truth, the performance of algorithms will be estimated with some degree of error. This error may be minimized with the incorporation of more annotations. One strategy to allocate crowds is therefore to do it such that the uncertainty in the evaluation of technology is minimized.
2. **Training.** One way to improve Machine Learning algorithms is to provide them with more training data of the kind where they tend to fail or where they show biased behavior. A second strategy to allocate crowds is therefore to do it such that the new annotations will help develop better technology by targeting critical training instances.
3. **Enrichment.** Large-scale music description may only be achieved with automatic algorithms. However, they are of course fallible and sometimes make erroneous or low-confidence predictions. A third strategy to allocate crowds is therefore to do it such that the new annotations will reduce the uncertainty in the music descriptors offered to end users.

Each of the above three objectives determines a priority to decide which music items are more important and should be annotated first.

3. TROMPA System Evaluation

Throughout this document we will follow a K-class classification task to exemplify the hybrid crowd-machine framework. Following Figure 2.1, let us assume only one system to solve this task. Let $y_i \in \{1,...,K\}$ be the class predicted for item i, and let $a_i \in \{1,...,K\}$ be the ground truth annotation for the same item.

3.1 Traditional Evaluation

The most straightforward and often used classification metric is accuracy, that is, the fraction of items whose class is correctly classified. However, in the presence of class imbalance it is more useful to evaluate on a class per class basis. The precision of class k is defined as the fraction of predictions of that class that are actually correct:

$$prec_k = \frac{\sum_i \mathrm{I}(y_i = k = a_i)}{\sum_i \mathrm{I}(y_i = k)} \tag{1}$$

where $I(\bullet)$ is the boolean indicator function. Precision is a measure of the noise when predicting a class. How exhaustive the system is for a class k can be measured with recall, which is similarly defined as the fraction of items annotated as k that are correctly predicted as such:

$$rec_k = \frac{\sum_i \mathrm{I}(y_i = k = a_i)}{\sum_i \mathrm{I}(a_i = k)} \tag{2}$$

Precision and recall are inversely related in practice, and one is typically high when the other is low. The F-score of class k is the harmonic mean of precision and recall, which is often used as a way to balance the two into a single metric:

$$f_k = 2\frac{prec_k \cdot rec_k}{prec_k + rec_k} \tag{3}$$

To give a single-score measure of performance, the average metric score over the K classes is usually computed (ie. the macro-average):

$$m = \frac{1}{K}\sum_k m_k \tag{4}$$

where m_k is a metric like precision, recall or F-score of class k. Based on these average scores, decisions are made as to the performance of systems and their further development.

3.2 Probabilistic Annotations

In order to realize the evaluation framework outlined in Figure 2.2, we need a way to model unknown annotations; random variables are used for this. Let A_i be a random variable representing the annotation for input item i. The distribution of this variable depends on the task: it would be a Bernoulli ($K = 2$) or Categorical ($K > 2$) random variable representing the probability of each class. Every such random variable can be defined in terms of $K - 1$ parameters from which we can compute some form of expectation and variance loosely representing our best guess for the annotation and its uncertainty. These parameters are the probabilities for each class, that is, $p_{ik} = P\left(A_i = k\right)$, where $\sum_k p_{ik} = 1$. Expectation and variance are defined per class k:

$$\mathrm{E}[A_{ik}] = p_{ik} \tag{5}$$

$$\mathrm{Var}[A_{ik}] = p_{ik}(1 - p_{ik}) \tag{6}$$

Whenever an annotator, expert or otherwise, provides an annotation a_i for item i, we can update $\mathrm{E}[A_{ik}] \leftarrow 1$ for $a_i = k$, $\mathrm{E}[A_{ik}] \leftarrow 0$ for $a_i \neq k$, and $\mathrm{Var}[A_{ik}] \leftarrow 0$ for all k, meaning that there is no uncertainty about the annotation anymore.

3.3 Probabilistic Performance Metrics

Because annotations are now represented by random variables, performance metrics need to be formulated in terms of random variables as well. This way, a metric score will be defined over a distribution of possible annotations to input items, and the uncertainty in the estimation of annotations will be propagated to the estimation of metric scores.

For simplicity, let $y_{ik} = \mathrm{I}(y_i = k)$ and $a_{ik} = \mathrm{I}(a_i = k)$. Precision (1), Recall (2) and F-score (3) can then be expressed as sums of products as follows:

$$prec_k = \frac{\sum_i y_{ik} a_{ik}}{\sum_i y_{ik}} \tag{7}$$

$$rec_k = \frac{\sum_i y_{ik} a_{ik}}{\sum_i a_{ik}} \tag{8}$$

$$f_k = 2\frac{\sum_i y_{ik} a_{ik}}{\sum_i y_{ik} + a_{ik}} \tag{9}$$

Substituting a known annotation a_{ik} with its corresponding random variable A_{ik}, we have the following probabilistic definition of performance metrics:

$$PREC_k = \frac{\sum_i y_{ik} A_{ik}}{\sum_i y_{ik}} \tag{10}$$

$$REC_k = \frac{\sum_i y_{ik} A_{ik}}{\sum_i A_{ik}} \tag{11}$$

$$F_k = 2\frac{\sum_i y_{ik} A_{ik}}{\sum_i y_{ik} + A_{ik}} \tag{12}$$

Because y_{ik} are constants, the expectation and variance for precision are easily computed as a sum as well:

$$\mathrm{E}[PREC_k] = \frac{\sum_i y_{ik}\mathrm{E}[A_{ik}]}{\sum_i y_{ik}} = \frac{\sum_i y_{ik} p_{ik}}{\sum_i y_{ik}} \tag{13}$$

$$\mathrm{Var}[PREC_k] = \frac{\sum_i y_{ik}\mathrm{Var}[A_{ik}]}{\left(\sum_i y_{ik}\right)^2} = \frac{\sum_i y_{ik} p_{ik}(1-p_{ik})}{\left(\sum_i y_{ik}\right)^2} \tag{14}$$

The formulation for recall is not as straightforward because it entails the division of two dependent random variables. An approximation using Taylor series expansion is possible [U13], but for simplicity we will consider the sum in the denominator as non-random [CAS06]:

$$\mathrm{E}[REC_k] \approx \frac{\sum_i y_{ik}\mathrm{E}[A_{ik}]}{\sum_i A_{ik}} = \frac{\sum_i y_{ik} p_{ik}}{\sum_i p_{ik}} \tag{15}$$

$$\mathrm{Var}[REC_k] \approx \frac{\sum_i y_{ik}\mathrm{Var}[A_{ik}]}{\sum_i {A_{ik}}^2} = \frac{\sum_i y_{ik} p_{ik}(1-p_{ik})}{\left(\sum_i p_{ik}\right)^2} \tag{16}$$

A similar approximation is followed for the F-score, which leads to:

$$\mathrm{E}[F_k] \approx 2\frac{\sum_i y_{ik} p_{ik}}{\sum_i y_{ik} + p_{ik}} \tag{17}$$

$$\mathrm{Var}[F_k] \approx 4\frac{\sum_i y_{ik} p_{ik}(1-p_{ik})}{\left(\sum_i y_{ik} + p_{ik}\right)^2} \tag{18}$$

The above definitions allow us to compute estimates of performance for a class. In order to estimate performance over K classes, we may compute expectation and variance as in (4):

$$\mathrm{E}[M] = \frac{1}{K}\sum_k \mathrm{E}[M_k] \tag{19}$$

$$\mathrm{Var}[M] = \frac{1}{K^2}\sum_k \mathrm{Var}[M_k] \tag{20}$$

where M_k is one of $PREC_k$, REC_k or F_k as defined above. Under the usual caveats of following the Central Limit Theorem, we can finally compute a confidence interval around the expected value to nicely account for the variance in the performance estimates. Specifically, a $100(1-2\alpha)\%$ interval can be computed as

$$\mathrm{E}[M] \pm z_\alpha \sqrt{\mathrm{Var}[M]} \tag{21}$$

where z_α is the α-th quantile of a Standard Normal distribution. For a typical 95% confidence interval, z_α is therefore 1.96 as usual.

4. Annotation Priority

Evaluation and training of systems, as well as the final corpus enrichment, is expected to improve as more annotations are made by the crowd. Following Figure 2.2, as more annotations are gathered in phase 1, more training data there is to develop systems in phase 2. Likewise, more annotations have a twofold impact on phase 3: the number of instances pending annotation is smaller and therefore human enrichment coverage is higher, and more training data is available to train the combination model. Annotation estimates from the combination model will be more accurate and precise, which has a similar impact on the system evaluation in phase 5.

A natural question to ask is therefore which instances should be annotated first. Without loss of generality, we can compute a weight ω_i for every instance i pending annotation, such that instances with higher weight have priority. When a user connects to the corresponding annotation tool as described in Deliverable D5.5 - "Annotation Tools", they will be assigned instances in descending order by weight, possibly tailored to their skills as modeled in Deliverable D4.2 - "Annotator Properties". Following the three annotation objectives described in Section 2, weights may be formulated differently.

4.1 Evaluation

For the purposes of evaluating systems, we have to note that the impact of a new annotation will be different for different metrics. For instance, a system that predicts k_1 would not see its $prec_1$ decreased if the annotation was k_2. However, rec_2 would decrease because it was expected to predict k_2. If the annotation was k_1 instead, both $prec_1$ and rec_1 would increase by different amounts. Looking at the impact of the annotation over classes, we can define a general weight as the expected difference between the annotation being from one class or not:

$$\omega_i^M = \left| \frac{1}{K} \sum_k \omega_i^{M_k} \right| \tag{22}$$

$$\omega_i^{M_k} = \mathrm{E}[M_k | A_{ik} = 1] - \mathrm{E}[M_k | A_{ik} = 0] \tag{23}$$

Following Precision as an example, the expected value conditioned on a certain value for A_{ik} can be decomposed into the effect of item i and every other item $j \neq i$:

$$\mathrm{E}[PREC_k | A_i = a_i] = \frac{y_{ik} a_i + \sum_{j \neq i} y_{jk} \mathrm{E}[A_{jk}]}{\sum_j y_{jk}} \tag{24}$$

$$\omega_i^{PREC_k} = \frac{y_{ik} + \sum_{j \neq i} y_{jk} \mathrm{E}[A_{jk}]}{\sum_j y_{jk}} - \frac{\sum_{j \neq i} y_{jk} \mathrm{E}[A_{jk}]}{\sum_j y_{jk}} = \frac{y_{ik}}{\sum_j y_{jk}} \tag{25}$$

As mentioned above, a new annotation of class k only has an effect if the system indeed predicted k (ie. $y_{ik} = 1$), and this effect would be inversely proportional to the number of positives. The case of Recall is more complicated because the denominator depends on the new annotation as well:

$$\mathrm{E}[REC_k|A_i=a_i]=\frac{y_{ik}a_i+\sum_{j\neq i}y_{jk}\mathrm{E}[A_{jk}]}{a_i+\sum_{j\neq i}\mathrm{E}[A_{jk}]} \tag{26}$$

$$\omega_i^{REC_k}=\frac{y_{ik}+\sum_{j\neq i}y_{jk}\mathrm{E}[A_{jk}]}{1+\sum_{j\neq i}\mathrm{E}[A_{jk}]}-\frac{\sum_{j\neq i}y_{jk}\mathrm{E}[A_{jk}]}{\sum_{j\neq i}\mathrm{E}[A_{jk}]} \tag{27}$$

Indeed, in the case of a true positive the first term will increase, leading to a positive effect, but in the case of a false positive its denominator will increase and lead to a negative effect. Finally, for the F-score the weight of an item is computed as follows:

$$\mathrm{E}[F_k|A_i=a_i]=2\frac{y_{ik}a_i+\sum_{j\neq i}y_{jk}\mathrm{E}[A_{jk}]}{a_i+y_{ik}+\sum_{j\neq i}y_{jk}+\mathrm{E}[A_{jk}]} \tag{28}$$

$$\omega_i^{F_k}=2\frac{y_{ik}+\sum_{j\neq i}y_{jk}\mathrm{E}[A_{jk}]}{1+y_{ik}+\sum_{j\neq i}y_{jk}+\mathrm{E}[A_{jk}]}-2\frac{\sum_{j\neq i}y_{jk}\mathrm{E}[A_{jk}]}{y_{ik}+\sum_{j\neq i}y_{jk}+\mathrm{E}[A_{jk}]} \tag{29}$$

where, similarly to Recall, effects may be both positive or negative depending on whether the system predicted class k or otherwise.

Computing weights over classes, as in (22), or over the set of systems being evaluated, the overall weight might thus stay close to zero. However, we note that weights may be computed targeting a specific class, or a specific system if so desired.

4.2 Training

For the purposes of improving the systems through better training data, we should target items that are expected to help their training processes. The off-diagonal elements of the classification confusion matrix can help us determine where this help is needed most. For instance, a system with low recall in class k_3 will benefit from more instances of such class, provided that it did not already predict them. In particular, if the system shows a bias for some other class k_2, we should target instances where $y_{i2}=1$ and $\arg\max_k p_{ik}=3$. In the unfortunate event that the annotation turns out to be some class other than k_3, at least this will help debias the combination model.

As mentioned above, one may target specific classes or systems, if so desired, by considering only certain columns and rows of the confusion matrices.

4.3 Enrichment

For the purposes of enrichment, we may prioritize items for which the uncertainty of the combination model is highest. One way to quantify this uncertainty is the entropy of the predicted distribution over classes:

$$\omega_i=-\sum_k p_{ik}\log p_{ik} \tag{30}$$

Alternatively, we can consider cases where the probability of the most likely class is low, regardless of the others classes:

$$\omega_i=1-\max_k p_{ik} \tag{31}$$

Iteratively, these criteria will maximize the certainty in the final enrichment. Note that, unlike the other cases, these weights are independent of the system predictions because they do not involve their development or their evaluation. Specific classes could be targeted though, if we consider instances where entropy is high (ie. with high uncertainty), and the class is the second most likely according to the combination model.

5. Combination Model

As presented in Figure 2.2, a combination model c is iteratively trained to predict human descriptions of the items pending annotation.

5.1 Feature Sets

The features used by this model may be described across two dimensions:

- ❖ **Internal vs. external.** Internal features are those directly related to the current task, that is, the outputs from the current set of systems and the known annotations. External features are not directly related, such as metadata, linked data, or descriptions made by other types of systems.
- ❖ **Static vs. dynamic.** Static features are those that do not change during the annotation process, such as the output from systems. Dynamic features are those that are updated as more annotations are known, such as the precision of one specific algorithm or annotator.

In principle, a model using a richer set of features should be able to make better predictions because it is more likely to find relationships between features and outcomes. This is particularly important for features of identifying nature such as some metadata, as they may lead to very strong relationships that systems would not be able to find by themselves. For example, that two composers are similar is very difficult to figure out algorithmically based only on the musical content of their work, but a couple known annotations (ie. dynamic), together with their identities (ie. external) as stated in metadata, can be leveraged to provide very strong priors on specific unknown annotations [US13]. For the problem of tailoring existing systems to TROMPA repositories, this is especially appealing.

However, such features may not always be available for a variety of reasons. For example, some of these features may require an analysis of the audio signal, which is not always available. Some others may relate to editorial metadata that could be missing. Similarly, some of these features may come from the output of other types of systems whose output is still unknown for part of the corpus. Therefore, the combination model should conform to a multi-tier structure in which the most complex model is at the top tier and the simplest one at the bottom. When a new annotation prediction is to be made, the most complex model is used if all features are available. If not, the next-tier model is subsequently tried. In the event that none of those models can be run for lack of features, a model based solely on internal static features is always available at the bottom of the tier structure. In the event of errors or other kinds of edge situations, the estimation process can always fall back to a non-informative prediction in which all classes are equally likely.

Figure 5.1 presents a sample configuration of a combination model. Which models go in which tier, is determined by their performance on the part of the corpus with known annotations, and this order may be updated as more annotations are added and the models get to be re-trained.

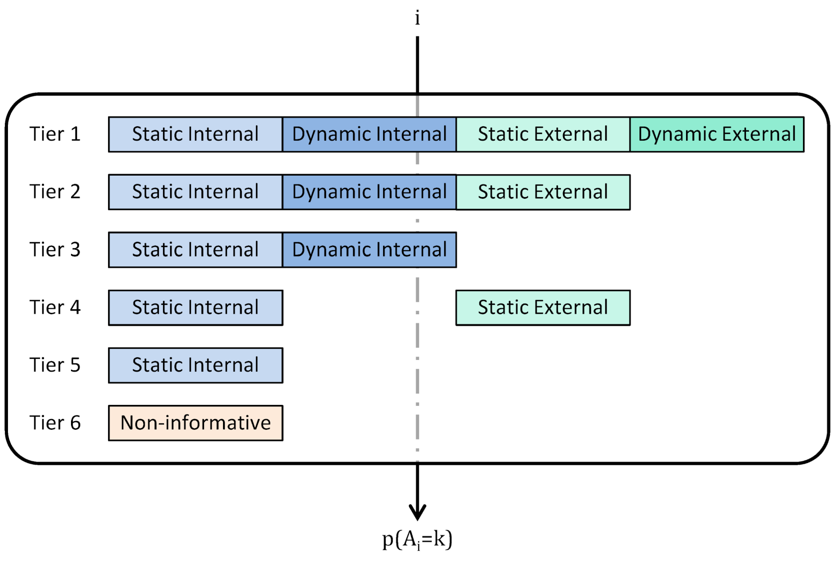

Figure 5.1. Sample configuration of a multi-tier combination model.

5.2 Features

For the specific case of K-class classification, we envision the following generic features:

- Static External Features (`se`)
    - `se_<p>.all.<v>`: probability of value `<v>` for property `<p>`.
    - `se_<p>.pred.value`: predicted value for property `<p>`.
    - `se_<p>.pred.prob`: probability of `se_<p>.pred.value`.
- Static Internal Features (`si`)
    - `si_<s>.<c>`: probability of class `<c>` as predicted by system `<s>`.
    - `si_<s>.pred.value`: class predicted by system `<s>`.
    - `si_<s>.pred.prob`: probability of `si_<s>.pred.value`.
    - `si_all.<c>.min`: minimum probability of class `<c>` as predicted by all systems.
    - `si_all.<c>.mean`: mean probability of class `<c>` as predicted by all systems.
    - `si_all.<c>.max`: maximum probability of class `<c>` as predicted by all systems.
    - `si_all.pred.value`: class with highest mean probability across all systems (ie. the majority vote).
    - `si_all.pred.prob`: mean probability of `si_all.pred.value`.
    - `si_all.pred.agree`: (int) number of systems that predict `si_all.pred.value`.
- Dynamic External Features (`de`)
    - `de_<p>.<c>.prec`: precision of property `<p>` for predicting class `<c>`.
    - `de_<p>.<c>.rec`: recall of property `<p>` for predicting class `<c>`.
    - `de_<c>.prec`: precision of the combination of all properties' values for predicting class `<c>`.
    - `de_<c>.rec`: recall of the combination of all properties' values for predicting class `<c>`.
- Dynamic Internal Features (`di`)
    - `di_<s>.pred.prec`: precision of system `<s>` when predicting `si_<s>.pred.value`.
    - `di_<s>.pred.rec`: recall of system `<s>` when predicting `si_<s>.pred.value`.
    - `di_<s>.<c>.prec`: precision of system `<s>` when predicting class `<c>`.

- `di_<s>.<c>.rec`: recall of system `<s>` when predicting class `<c>`.
- `di_all.pred.prec`: precision of the majority vote when predicting `si_all.pred.value`.
- `di_all.pred.rec`: recall of the majority vote when predicting `si_all.pred.value`.
- `di_all.<c>.prec`: precision of the majority vote when predicting class `<c>`.
- `di_all.<c>.rec`: recall of the majority vote when predicting class `<c>`.
- `di_<c>.prec`: precision of the combination of all systems' predictions for predicting class `<c>`.
- `di_<c>.rec`: recall of the combination of all systems' predictions for predicting class `<c>`.

where a *property* is an arbitrary description of the input item that may be expressed in one of a number of categories, such as *relaxed*, *sad*, *acoustic*, etc. These features are of course not exhaustive and may be modified to suit the needs of specific tasks.

5.3 Model Families

The implementation of each of the models at different tiers may be achieved through a variety of classification algorithms, provided that they make probabilistic predictions, that is, that they are able to predict the probability of each class as opposed to just the most likely class. Although by no means exhaustive, in TROMPA we consider the following alternatives:

- Uniform distribution (`uniform`). This model serves as a non-informative baseline in which every class is equally likely.
- Empirical distribution (`empirical`). This model serves as an informative baseline in which the probability of each class is the same as observed in the known annotations.
- Multinomial Logistic Regression (`multinom`) [VR02].
- Classification Tree (`tree`) [BFOS84].
- Support Vector Machine (`svm`) [CV95].
- Random Forest (`rf`) [B01].

6. Sample Application: Music Emotion

To illustrate the TROMPA evaluation methodology, here we present two experiments for the task of emotion recognition, which links directly to the Music Enthusiasts use case. We follow the traditional dimensional approach for describing emotion. This approach considers emotion in a two dimensional space comprising valence (pleasantness or positiveness) and arousal (energy or activation) [R80]. From this space, annotations are typically at the quadrant level:

- **Q1:** positive valence and positive arousal.
- **Q2:** negative valence and positive arousal.
- **Q3:** negative valence and negative arousal.
- **Q4:** positive valence and negative arousal.

The following subsections describe the data and systems used in the experiments, followed by the first experiment about the combination model, and the second experiment about system evaluation.

6.1 Data and Systems

With respect to musical data and annotations, we use the following three standard datasets:

- **4Q Dataset**. A collection of 900 music clips and tags collected from the AllMusic API [PMP18]. Emotion tags were selected by intersecting them with Warriner's list [WKB13], and a subsequent manual blind validation was conducted by human subjects. Audio is available to compute external features.
- **MediaEval Database for Emotional Analysis in Music** (DEAM). A collection of 1,802 music clips gathered from the Free Music Archive, Jamendo and the MedleyDB database [AYS17]. Annotations were crowdsourced with Amazon Mechanical Turk, collecting 10 annotations per clip. Audio is available to compute external features.
- **CH818 Dataset**. A collection of 818 music clips annotated by three music excerpts using [-10,10] scales for valence and arousal [HY17]. Categorical annotations are computed by discretizing in positive and negative values. Audio is not available to compute external features.

The full dataset thus comprises 3,520 instances. The distribution of quadrant annotations per dataset is reported in Table 6.1. As can be seen, the dataset is unbalanced and has a clear bias for Q1 and Q3.

Dataset	Instances	Q1	Q2	Q3	Q4
4Q	900	225	225	225	225
DEAM	1,802	647	229	686	240
CH818	818	391	127	211	89
Total	3,520	1,263	581	1,122	554

Table 6.1. Total number of instances per dataset, and number of instances per class.

As for systems, we evaluate four different systems developed in the context of Deliverable D3.2 - "Music Description", which we name s1 to s4 for simplicity. All systems are based on denoising convolutional autoencoders in order to train arousal-valence classifiers exploiting speech data as pretraining. The classifiers are structured in a multi-task learning approach in order to predict arousal (positive or negative), valence (positive or negative), and quadrant. In addition, we consider a simple ensemble system that takes the majority vote of the probabilistic predictions of each of the four systems.

Table 6.2 reports the performance of all systems and the ensemble, for each class separately as well as the macro-average. We note several properties of these systems that make them specially suitable for this experiment. First there are clear biases: system s2 only predicts Q1, and s4 never predicts Q2. Second, there is no system consistently outperforming the others. For specific quadrants we see systems clearly standing out, but not in general. Third, the ensemble system performs best in some edge cases, but does not have good performance overall.

		s1	s2	s3	s4	ensemble
Q1	Precision	0.351	0.359	0.353	0.355	**0.360**
	Recall	0.359	**1**	0.470	0.754	0.798
	F-score	0.355	**0.528**	0.403	0.483	0.496
Q2	Precision	0.198	0	0.186	0	**0.221**
	Recall	0.136	0	**0.188**	0	0.072
	F-score	0.161	0	**0.187**	0	0.109
Q3	Precision	**0.322**	0	0.303	0.307	0.305
	Recall	**0.296**	0	0.089	0.228	0.114
	F-score	**0.308**	0	0.138	0.262	0.166
Q4	Precision	0.146	0	0.137	**0.500**	0.134
	Recall	0.209	0	**0.229**	0.002	0.027
	F-score	**0.172**	0	**0.172**	0.004	0.045
Average	Precision	0.254	0.090	0.245	**0.290**	0.255
	Recall	0.250	0.250	0.244	0.246	**0.253**
	F-score	**0.249**	0.132	0.225	0.187	0.204

Table 6.2. Performance of each of the four systems, as well as a simple ensemble model. Best per row in bold face.

6.2 Experiment 1: Combination Model

With respect to the external features to be used by the combination model, we compute a total of twelve general and high-level descriptors[1] from the audio signal using Essentia [BWGG13]:

- ❖ Danceability (yes, no).
- ❖ Mood features: 6 binary descriptors (acoustic, aggressive, electronic, happy, party, relaxed), and the 5 mood clusters of Hu and Downie [HD07].
- ❖ Gender (female, male).
- ❖ Timbre (bright, dark).
- ❖ Atonality (tonal, atonal).
- ❖ Instrumental (voice, instrumental).

For this experiment, each descriptor will be treated as a property of the music instance to be used as an external feature in the combination model (see Section 5.2). We note that these features can not be computed for the CH818 dataset because it does not provide the audio signal. Therefore, the combination model will only be able to use internal features when estimating annotations for this dataset (eg. in a model like the one in Figure 5.1, only tiers 3 and 5 would be possible for CH818).

In order to evaluate the performance of the combination model, we train it varying the number of available known annotations from 10 to 2,000 in increments of tens and hundreds (we do not use

[1] https://essentia.upf.edu/svm_models/accuracies_v2.1_beta1.html

CH818 here because it does not allow us to compute external features). We repeat 5 times with random selections of known annotations and compute the mean over replications. Figure 6.1 shows the F-score of the combination model using only internal features (left) and using both internal and external features (right), for each of the model families used internally by the combination model in its tier structure (see Section 5.3).

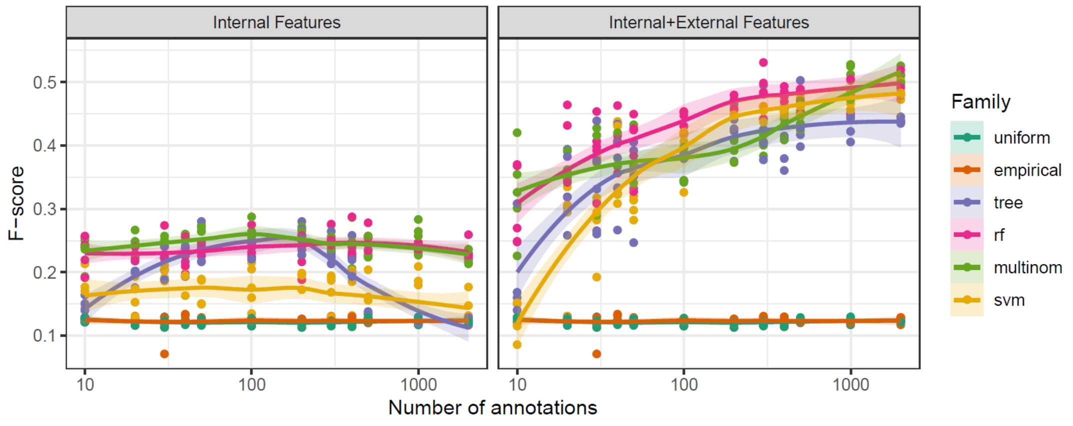

Figure 6.1. F-score of the combination model as a function of the number of known annotations. Each point represents one replication of the experiment.

We can first notice that the combination model performs better than the ensemble system (F=0.204) even if using only internal features, provided that families `rf` or `multinom` are used. When external features are used, all families outperform the ensemble system with as few as 10-20 known annotations. In particular, the `rf` family easily achieves twice as good performance. Family `svm` achieves very good performance provided that there are external features and a fair amount of known annotations. The poor performance of the baselines `uniform` and `empirical` highlight the bias in the class distribution.

Another aspect to evaluate the combination model is the certainty in its estimations. A model that is certain in its predictions will reduce variance in the evaluation and enrichment when it is correct, but will increase bias if it is incorrect. Figure 6.2 shows the entropy of the predictions from the combination models (high entropy means low certainty). We can see that the `multinom` family, which is among the best performing families with respect to F-score, especially when using only internal features, appears to be overconfident in its predictions. On the other hand, the `rf` family seems to behave well with respect to performance and uncertainty. This is due to how the random forests are built by resampling the training data, which offers a non-parametric way to account for sample variation. On the other hand, `svm` family performs notably well in terms of certainty, so it seems like a very good candidate when there are external features and some known data. Both `rf` and `svm` are particularly appealing because their certainty is not heavily affected by the amount of known annotations.

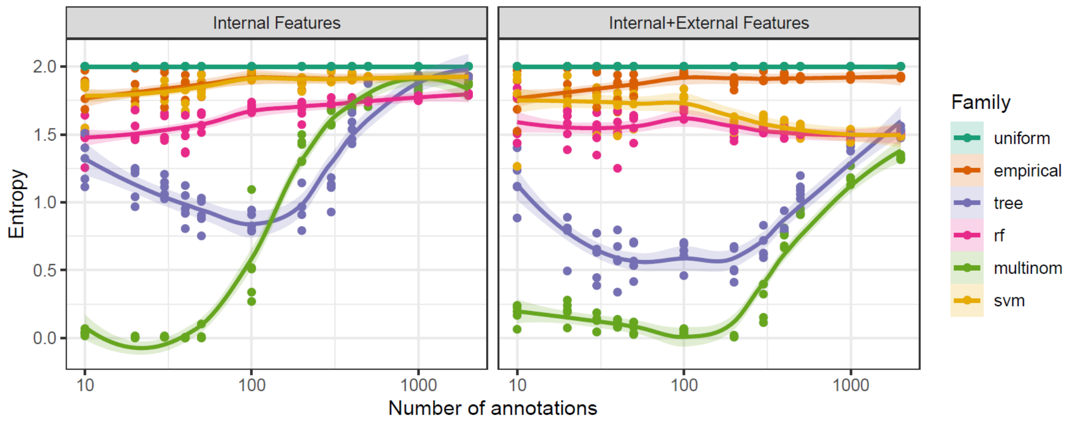

Figure 6.2. Entropy of the estimation of the combination model as a function of the number of known annotations. Each point represents one replication of the experiment.

As a way to explore possible biases in the combination models, Figure 6.3 shows the RMSE in their confusion matrices compared to that of a perfect model. We can see that `rf` and `multinom` are again the best performing families when only internal features are available. When there are external features too, `svm` also performs well provided again that there is a good amount of known annotations. In all cases though, we see that bias decreases as more data is available.

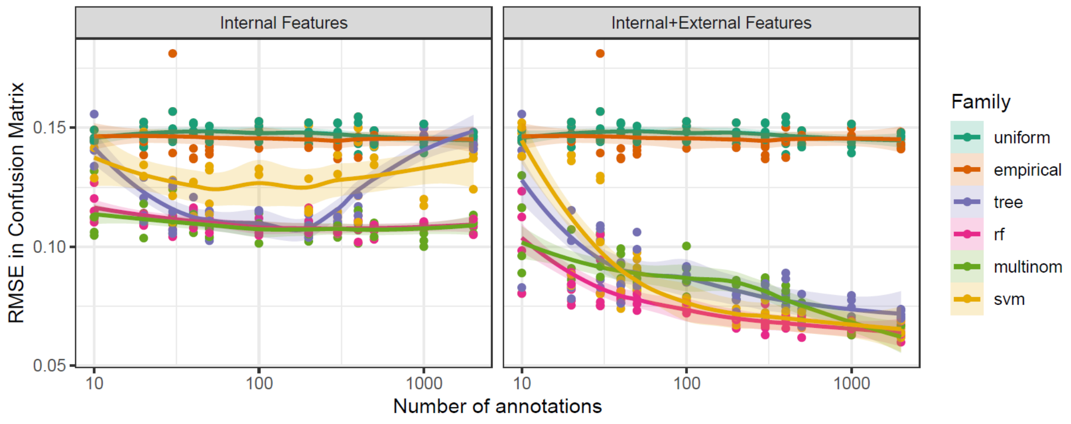

Figure 6.3. Bias of the combination model as a function of the number of known annotations. Each point represents one replication of the experiment.

Overall, these results suggest that a combination model implemented with random forests (`rf`) will perform best, while a model implemented with multinomial regression (`multinom`) will probably be overconfident and bias the evaluation results. A model implemented with support vector machines (`svm`) is likely to perform well once some annotations are available.

6.3 Experiment 2: System Evaluation

Here we report the results of similar experiments from the point of view of evaluating systems in the face of missing annotations. In particular, we simulate the TROMPA framework in Figure 2.2 to evaluate the four systems and enrich the corpus, given that no annotations are present. To start the process, we assume the campaign initiator is able to provide two instances from each class to fit a first combination model. In the case of emotion recognition, this means only 8 annotations. From that point on, the framework enriches the corpus using the predictions from the combination model, estimates system performance and establishes a priority for the next items to be annotated. It then iteratively waits for another 20 annotations to refit the combination model and repeat.

Figure 6.4 shows the results when using random forests in the combination model. Specifically, the figure shows the error in the estimation of F-scores for each class, and for the macro-averaged F-score. In average terms, we see that the estimation rapidly approaches zero error with a few dozen annotations and largely stays within a 5% error margin (dashed lines), but when the annotation set is close to empty the estimation is quite poor. Across classes we see clear differences largely due to the imbalance in the corpus. The variance in the predictions (grey ribbons) of course decreases as more annotations are made, and except when there are again very few annotations, they largely cover the estimation error as expected.

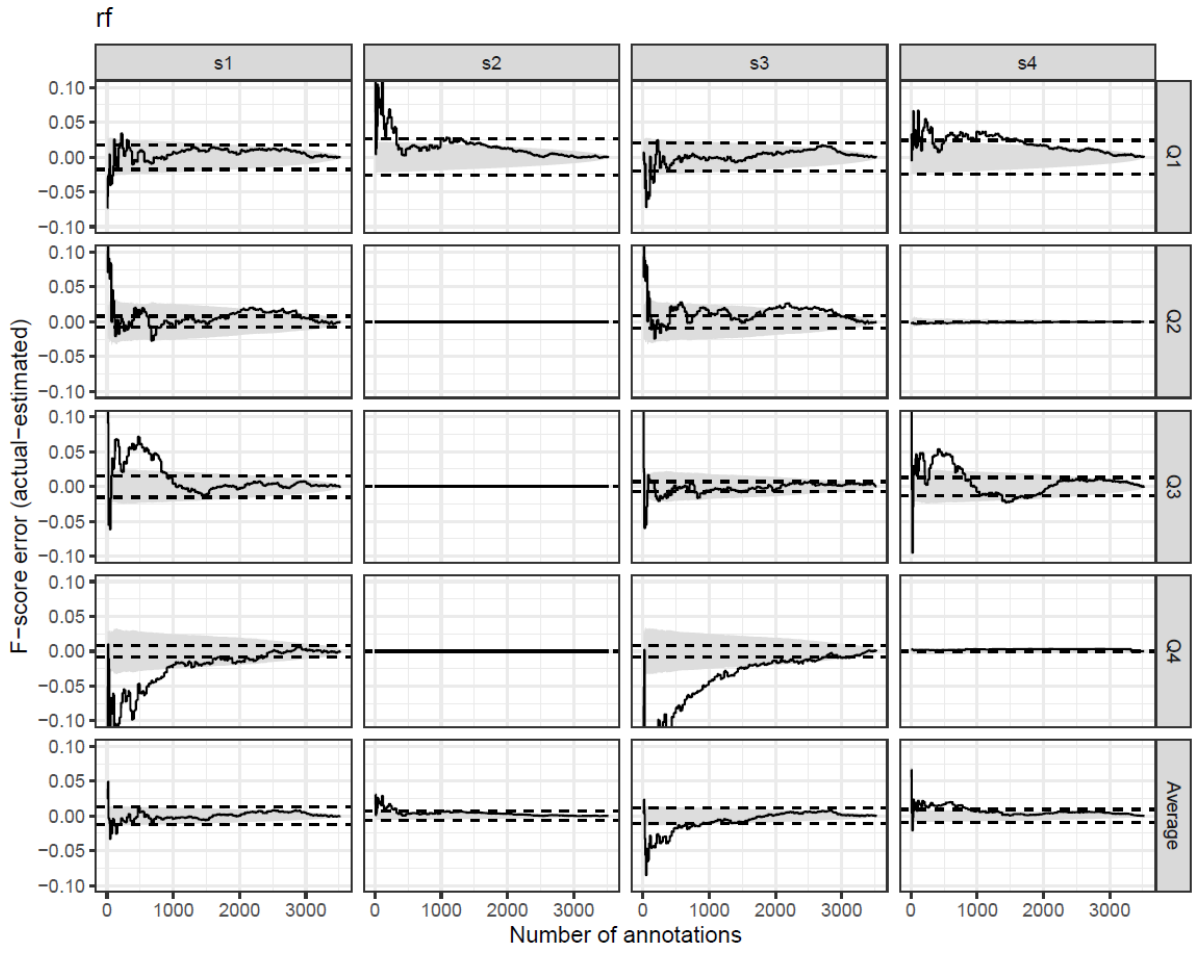

Figure 6.4. Error in the estimation of system performance (F-score), as a function of the number of known annotations, using random forests in the combination model. The gray ribbons represent a 95% confidence interval as per (21), and the dashed horizontal lines mark 5% error size.

For comparison, Figure 6.5 shows the same plots but when using multinomial regression in the combination model. There are two main differences. First, the estimation is more erratic with low numbers of annotations, but more importantly, the variance is largely underestimated as shown by the narrow ribbons. This is because the model is not presented with enough diversity when training, and only when there are dozens of annotations it then learns that it is making more mistakes. As mentioned above, random forests deal better with this issue because they resample the training data when fitting the model and, as a consequence, are also better prepared to avoid learning the class imbalance.

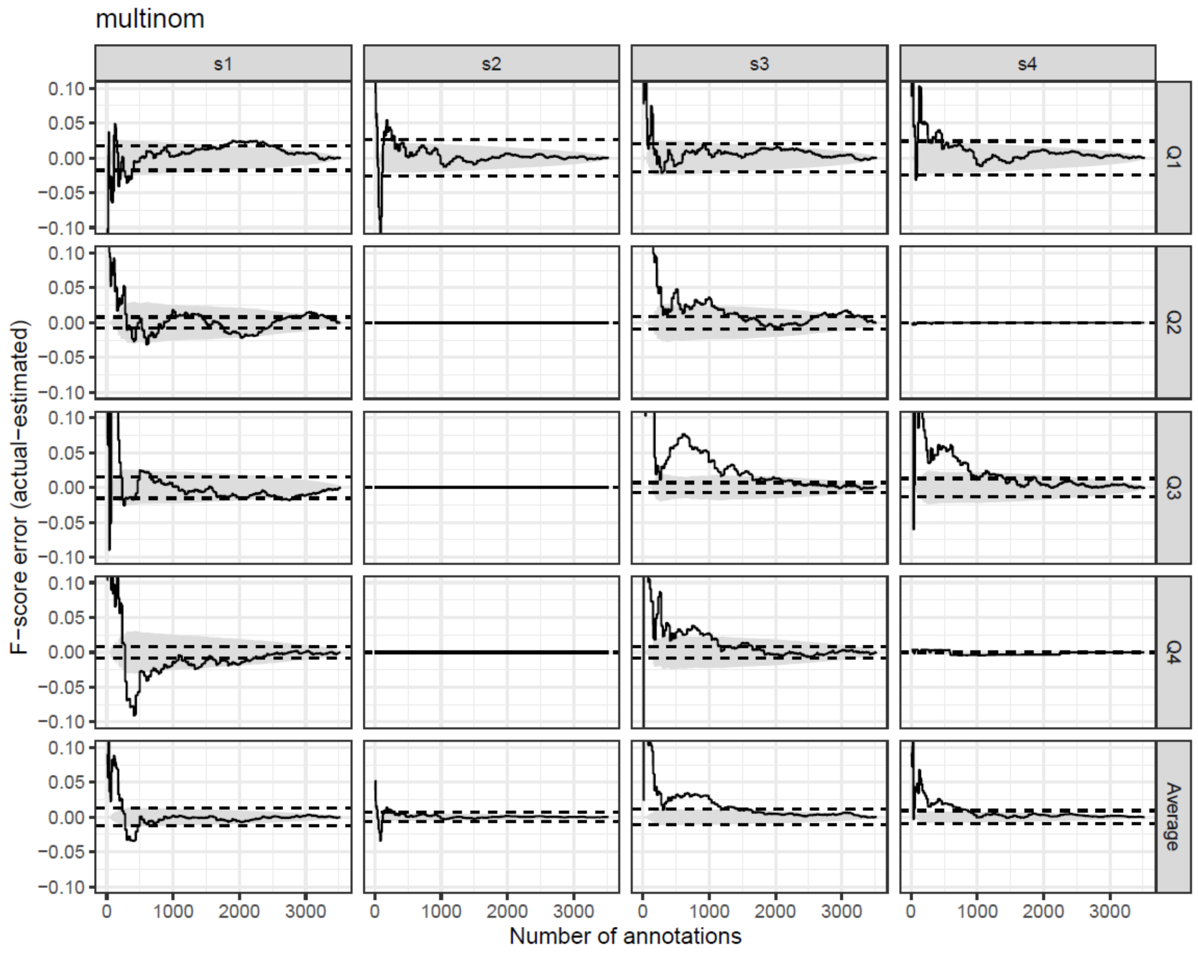

Figure 6.5. Same as Figure 6.4, but using multinomial regression for the combination model.

7. Conclusion

TROMPA presents unique challenges for the evaluation of music description systems because of the lack of ground truth data. This has direct impact not only on the development of technology, but also on their evaluation and final use to enrich classical music corpora. This deliverable described a hybrid crowd-machine framework for the continuous evaluation of WP3 technology by modeling uncertainty due to the missing ground truth data, and by using the work of the crowd where it is most beneficial. In doing so, this framework enables reliable and efficient development of technology tailored to the corpora considered in TROMPA, as well as improved enrichment of said corpora by combining crowds and algorithms.

Using the case of emotion recognition as an example, we have shown how the framework can successfully estimate system performance even when there are just a few dozen annotations, unbalanced datasets, and biased systems. Additionally, we showed that the final enrichment produced by this hybrid framework achieves twofold performance compared to a simple ensemble approach over the raw algorithm output.

Although no further versions of this deliverable are scheduled, research on the topics here described, as well as their integration in the Contributor Environment, will continue until the end of the project. Any contribution in the form of scientific publications or similar will be published on the website of the project and properly reported on the periodic reports.

8. References

8.1 Written References

[AYS17] Aljanaki A., Yang Y., and Soleymani M. (2017). Developing a benchmark for emotional analysis of music. PLoS ONE, 12(3).

[B01] Breiman L. (2001). Random Forests. Machine Learning, 45(1).

[BFOS84] Breiman L., Friedman J.H., Olshen R.A., and Stone C.J. (1984). Classification and Regression Trees. Wadsworth.

[BWGG13] Bogdanov D., Wack N., Gómez E., Gulati S., Herrera P., Mayor O., Roma G., Salamon J., Zapata J.R., and Serra X. (2013). Essentia: An Audio Analysis Library for Music Information Retrieval. International Society for Music Information Retrieval Conference.

[CAS06] Carterette B., Allan J., and Sitaraman R. (2006). Minimal Test Collections for Retrieval Evaluation. ACM SIGIR Conference on Research and Development in Information Retrieval.

[CV95] Corte C. and Vapnik V.N. (1995). Support-vector networks. Machine Learning, 20(3).

[HD07] Hu X. and Downie J.S. (2007). Exploring mood metadata: Relationships with genre, artist and usage metadata. International Conference on Music Information Retrieval.

[HY17] Hu X. and Yang Y. (2017). Cross-Dataset and Cross-Cultural Music Mood Prediction: A Case on Western and Chinese Pop Songs. IEEE Transactions on Affective Computing, 8(2).

[PMP18] Panda R., Malheiro R., and Paiva R.P. (2018). Musical Texture and Expressivity Features for Music Emotion Recognition. International Society for Music Information Retrieval Conference.

[R80] Russell J.A. (1980). A circumplex model of affect. Journal of Personality and Social Psychology, 39(6).

[U13] Urbano J. (2013). Evaluation in Audio Music Similarity. PhD Thesis.

[US13] Urbano J. and Schedl M. (2013). Minimal Test Collections for Low-Cost Evaluation of Audio Music Similarity and Retrieval Systems. International Journal of Multimedia Information Retrieval, 2(1).

[VR02] Venables W.N. and Ripley B.D. (2002). Modern Applied Statistics with S. Springer.

[WKB13] Warriner A.B., Kuperman V., and Brysbaert M. (2013). Norms of valence, arousal, and dominance for 13,915 English lemmas. Behavior Research Methods, 45(4).

8.2 List of abbreviations

Abbreviation	Description
RMSE	Root Mean Square Error
TUD	Technische Universiteit Delft
UPF	Universitat Pompeu Fabra