The CLAS system at the MediaEval 2014 C@merata Task

Stephen Wan
CSIRO
Sydney, Australia
Stephen.Wan@csiro.au

ABSTRACT

This paper describes the CLAS system which accepts natural language queries in the domain of music theory to perform passage retrieval from a musical score. This system was produced for participation in the C@merata MediaEval 2014 shared task. The system uses a domain-specific parser to interpret the query and answer generation methods based on feature unification. Performance on this task was encouraging with 0.76 precision and 0.96 recall.

1. INTRODUCTION

This paper describes the CLAS system which selects processes and retrieves potentially relevant answers from structured data given a natural language query. In this work, the queries and the structured data are in the domain of music theory, as defined by the C@merata 2014 task [1]. The CLAS system produces candidate answers by selecting passages from an musical score (in XML). Answers may be any consecutive time points spanning multiple whole and partial bars.

For example, a query ``4 crotchets" should retrieve any sequence of four consecutive elements in the score where each element is a note and each note has the time duration of a crotchet (one quarter of a whole note). In such a system, expert knowledge is needed to interpret the query. However, this not just limited to definitions of musical concepts (e.g., ``crotchet"). For example, the query ``4 crotchets" should be interpreted not just as any four notes with crotchet duration within the music (compare this to a general knowledge query ``4 composers" requiring any four musical composers to be provided) but specifically four notes in sequence. Furthermore, these four notes would typically be expected to be in the same *voice* or *part*; for example, if it were a piano score for two hands, the four crotchets might be a sequence written in the treble clef, played by the right hand.

In this paper, we describe a system that processes the input query, mapping from words in English to music metadata corresponding to the search criteria, or features, represented as a set of attribute-value pairs. An exhaustive search of an XML score is performed, note by note, for candidate answers using feature unification.

This system achieved an overall performance of 0.76 precision and 0.96 recall. The remainder of the paper outlines the system in more detail and presents the C@merata evaluation results.

2. APPROACH

The CLAS system interprets the natural language query (NLQ) to find candidate answer passages from the score. Briefly, the system:

1. pre-processes tokens and maps these to a list of concepts, or the concept representation (CR).
2. scans the CR and consumes concepts if they define the scope of the answer.
3. parses the remaining CR list to construct the query representation (QR), a sequence of feature structures that indicate the type of answer required, using handwritten parsing rules which implicitly capture the domain-specific interpretation of the NLQ.
4. Compares the QR with a subset of the data in the XML, referred to as the Scoped Data (SD), represented as a list of FS, from which candidate answers can be found using feature unification.

Copyright is held by the author/owner(s).
MediaEval 2014 Workshop, October 16-17, 2014, Barcelona, Spain.

2.1 Mapping Query Terms to Concepts

The system uses a handcrafted lexicon that maps from terms in the NLQ to concepts in the music theory domain, using the following five steps.

In Step 1, multi-word entities such as “down bow” are mapped to a single token “down_bow” to allow correct tokenisation. In Step 2, tokens such as “Vb”, denoting the dominant chord (“V”) in the first inversion (“b”), are separated into the two components. In Step 3, quotation marks are used tag quoted words as being lyrics (Note: the lexicon used here is limited to music theory terms only and does not include the wider language from which lyrics may originate). In Step 4, tokens are separated using whitespace as a delimiter. Finally, in Step 5, tokens are mapped to their conceptual form using the lexicon. Non-contentful words that are not used to construct the QR (e.g., the article “a” or redundant information about sequence order like “followed by”) are mapped to a null token and are thus ignored.

For example, the word "crotchet" is mapped to "_note:length.1", indicating that the word relates to a “note” FS, where the feature “length” takes the value “1”. Similarly, the word "quarter" (as in “quarter note”) is also mapped to this sense "_note:length.1".

Words can have multiple meanings. For example, the word "perfect" is mapped to "_sequence:int_quality.PERFECT; _chord_sequence:cadence.PERFECT", indicating two senses: one referring to the quality of an interval (e.g., “a perfect fifth) , or a type of chord sequence (e.g., “a perfect cadence”).

2.2 Building Scoped Data

The system labels each NLQ with a type T specifying the type of answer required and the scope of the XML data to be examined for an answer (i.e., the SD). In this work, we defined four types: (i) harmonic, (ii) cadence, (iii) style; and (iv) note. Each type specifies rules for: (1) converting from the XML representation into an SD; (2) parsing rules to convert the CR into a QR; and (3) candidate generation rules.

A scan of the CR is used to determine the type T by searching for concepts specifying the data “granularity”. If any are found, these are removed from CR and used to set the type. For example, “simultaneous”, as in “simultaneous second” (referring to an interval of a second where both notes are sounded concurrently), is mapped to the concept

"_data:granularity.HARMONIC", indicating the *harmonic* type. In this case, the SD is defined as a list of chordal notes, taken from a block chord view of the score.[1]

The *cadence* and *style* types also scope the data as a list of chords. If no other type is indicated by a concept in CR, the default *note* type is used, defining the SD as the concatenation of the sequence of notes in each voice.

For queries where the voice or clef is specified, for example "treble clef" or "soprano part", the corresponding concepts are used to filter the data to include just that voice.

2.3 Building a Query Representation (QR)

The remaining tokens in CR are used to create a list of FSs of type T following a bespoke rule-based parsing process. The CR is processed in reverse order (assuming head-final noun phrases) and FSs are constructed in a process loosely based on reduction in a shift-reduce parser.

For example, the query "a C sharp crotchet and a D minim" is mapped to the CR "[_note:name.C, _note:accidental.SHARP, _note:length.1, _note:name.D, _note:length.2]". The concepts "[_note:name.D, _note:length.2]" are consumed first and used to populate a FS. At this point, the "_note:length.1" concept is encountered. Because the current FS already has a note length value (a "minim"), the FS is popped off and pushed onto the QR list. A new FS is then used to consume the remaining tokens: "[_note:name.C, _note:accidental.SHARP, _note:length.1]". The CR is now empty and the QR is a list of two FSs corresponding to the notes. Parsing works similarly for the other types. For example, cadences are sequences of chord FSs.

2.4 Matching a Query Representation to Scoped Data

Once a QR is generated, the SD sequence is then iterated through and at each position a match to the QR is attempted using feature unification. If a match is found, then a candidate answer passage is stored.

For style answers, a different process is used based on simple heuristics. For example, the *homophony* and *polyphony* answer generation processes consider chords for passing notes, indicated by implicit ties. Consequently, the QR for this type is an empty list since no feature unification takes place.

3. RESULTS AND DISCUSSION

3.1 Results

Performance for this system is encouraging. The overall results are presented in **Table 1**, which lists the recall and precision for answers at two granularities of answers: the correct bars and also the correct beats. Considering the hand-crafted lexicon and the bespoke parsing mechanism, the system performs reasonably well at both granularity answer types, with precision around 0.7 and recall at around 0.9. At the time of writing, the average performance of systems participating in the C@merata task is not available.

The C@merata evaluation also provides additional statistics regarding performance based on the type of query. The system does well with queries related to the properties of notes in a sequence. For these categories, "simple pitch" (e.g., "G"), "simple length" (e.g., "quarter note rest"), "pitch and length" (e.g., "half note C"), "expression" (e.g., "fermata A natural"), precision and recall is above 0.86. Indeed in some cases, recall and precision is 1.0.

The general approach of creating sequences of feature structures (the "followed by" query type, e.g., "quaver C# followed by crotchet B" performed reasonably, with precision of 0.748 and recall of 0.859 for the beat answer types (performance increases for the bar answer type). From this, we infer that the general assumptions underpinning the way noun phrases about notes are transformed into the query representations using the reduction process are sound.

Granularity	Precision	Recall
Beat	0.713	0.904
Bar	0.764	0.967

Table 1. Overall Results

3.2 Future Work

In this work, time constraints affected the choice of methods used in the CLAS system. For example, instead of the bespoke parsing process used here to map from the query tokens to the feature structures in the Query Representation, an alternative method might be to create a context-free grammar for the domain sublanguage and to use a tool like NLTK[2] to parse the tokens, resulting in a syntactic parse. This linguistic structure can then be mapped to the feature structures. In future work, we will examine the parsing of noun phrase structures in which the features for matching are propagated up to an appropriate node in the tree. These can then be collected to form the Query Representation.

Finally, instead of enumerating exhaustively through all notes, in future work, we will examine the use of search engines to find candidate starting positions, from which feature unification processes can then start. In this approach, notes might be treated as quasi-documents, allowing them to be indexed by metadata based on musical properties.

4. CONCLUSION

In this work, expert knowledge in music theory was directly incorporated into a bespoke parser and lexicon. These were used to interpret a music NLQ, and a scoping process to reduce the space for candidate answers. Parsing was performed using a reduce-style process. Matches were performed using feature unification. Performance on this task was encouraging with 0.76 precision and 0.96 recall.

5. REFERENCES

[1] Sutcliffe, R., Crawford, T., Fox, C., Root, D.L., and Hovy, E. 2014. The C@merata Task at MediaEval 2014: Natural language queries on classical music scores. In *MediaEval 2014 Workshop*, Barcelona, Spain, October 16-17 2014.

[1] The method chordify from the *music21* package (http://web.mit.edu/music21/) is used to produce this view.

[2] http://www.nltk.org/