UNIVERSITAT POLITÈCNICA DE VALÈNCIA
DEPARTMENT OF COMPUTER SYSTEMS AND COMPUTATION
MASTER'S THESIS

UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

Integration of Machine Learning and Language Processing Technologies into Video Lecture Platforms.

Master in Artificial Intelligence, Pattern Recognition and Digital Imaging.

Alejandro Pérez González de Martos

Directed by:
Dr. Alfons Juan Ciscar
Dr. Jorge Civera Saiz

September 11, 2014

I would like to thank the UPV's Machine Learning and Language Processing group and the Jožef Stefan Institute for making this thesis possible.

Contents

Chapter 1

Introduction

1.1 Motivation

Online multimedia repositories are rapidly growing and becoming evermore consolidated as key knowledge assets. This is particularly true in the area of education, where large repositories of video lectures are being established on the back of increasingly available and standardised infrastructures. Well-known examples of this include massive open online courses (MOOCs) aggregators such as Coursera [1], the *Universitat Politènica de València* (UPV) poliMedia platform system for the cost-effective ellaboration and publication of quality educational videos [33], and VideoLectures.NET, an award-winning free and open access educational video lectures repository [47].

As in other repositories, transcription and translation of video lectures in platforms such as poliMedia and VideoLectures.NET is needed to make them accessible to speakers of different languages and to people with disabilities [14, 48]. Moreover, transcriptions and translations of video lectures can also be helpful in the development of multiple digital content management applications such as lecture categorisation, summarisation, recommendation, automated topic finding or plagiarism detection.

However, most lectures in these platforms are neither transcribed nor translated because of the lack of efficient solutions to obtain them at a reasonable level of accuracy. This was the motivation behind the transLectures European project [38, 45], which aimed at developing innovative, cost-effective solutions for producing accurate transcriptions and translations for large video repositories.

1.2 Scientific and technical goals

This work aims to effectively integrate the tools developed in transLectures [44, 41] into different video lecture platforms. In particular, these tools will be integrated into the aforementioned poliMedia and VideoLectures.NET repositories. In the case of the latter, video lecture transcriptions will be used to develop a lecture recommender application as part of the PASCAL Harvest Project La Vie [32]. In addition, transLec-

tures solutions will also be integrated into Opencast Matterhorn [22], an open-source platform to support the management of educational multimedia content adopted by more than 40 different organisations all around the world.

It should be noted that this thesis is framed within the research project transLectures, and it is therefore part of a collaborative work. That said, special attention will be paid to the author's individual contributions.

1.3 Document structure

This thesis is organised as follows: Chapter 2 introduces the machine learning and language processing computer science fields, focusing on the automatic speech recognition and statistical machine translation tasks. In this chapter, the poliMedia and VideoLectures.NET video lecture repositories are also described, as well as the Opencast Matterhorn platform. Next, the set of tools for the integration of transLectures technologies into video lecture repositories is presented in Chapter 3. Chapter 4 addresses the integration of these tools into the Opencast Matterhorn platform. Chapter 5 describes a lecture recommender system that uses automatic speech transcriptions to better represent lecture contents at a semantic level. In particular, this system was implemented for the VideoLectures.NET repository. Finally, concluding remarks are given in Chapter 6.

Chapter 2

Preliminaries

2.1 Machine Learning and Language Processing

The Machine Learning field, evolved from the broad field of Artificial Intelligence, deals with building computer systems that optimise performance criteria using previous data or experience. It involves the development of mathematical algorithms that discover knowledge from specific data sets, and then "learn" from the data in an iterative fashion that allows predictions to be made. Today, Machine Learning includes a variety of applications such as natural language processing, search engine function, medical diagnosis, computer vision, and stock market analysis.

As the reader might guess from the aforementioned applications, the range of learning problems is clearly large. To avoid reinventing the wheel for every new Machine Learning application, the research community has tended to formalise the problems in a set of fairly narrow prototypes. Machine Learning systems can be classified along three particularly meaningful dimensions [10]:

- Classification on the basis of the *underlying learning strategies* used.
- Classification on the basis of the *knowledge representation*.
- Classification in terms of the *application domain* of the performance system for which knowledge is acquired.

Each point in the space defined by the above dimensions corresponds to a particular learning strategy. In this thesis, we concentrate on the *classification* task, also referred to as *pattern recognition*, where one attempts to build algorithms capable of automatically constructing methods for distinguishing between different exemplars, based on their differentiating patterns. More specifically, we will focus on the particular tasks of speech recognition and machine translation, which can be included in the Natural Language Processing (NLP) field. The NLP handles the research for efficient methods to enhance human-machine (and human-human) communication.

In the following sections we give a short overview of the statistical pattern recognition approach, and briefly introduce the automatic speech recognition and machine translation problems.

2.1.1 Pattern recognition

The term *pattern recognition* refers to the task of placing some object to a correct class based on the measurements about the object [43]. Concretely, pattern recognition systems aim to recognise their environment from data acquired by appropriate sensors (optical, acoustic, thermal, chemical, etc.). Some of the specific pattern recognition objectives include speech and handwriting recognition, machine translation, fingerprint/facial vertification, and disease identification. In order to concisely describe the different pattern recognition problems, probability theory has proven to be the most adequate language. We will assume the reader has some prior knowledge on probability theory, and therefore some basic definitions will be skipped. For more details and a very gentle and detailed discussion, see the book of *Introduction to probability* [16].

According to the classification paradigm, to recognise means to classify into one out of C given classes or categories with minimum probability of error. Many pattern recognition systems can be thought to consist of five stages:

1. **Sensing**, which refers to the measurement or observation of the object to be classified.

2. **Pre-processing**, which is the process of filtering the raw data for noise supression and other operations to improve the quality of the data.

3. **Feature extraction**, which refers to the process of extracting relevant data for the classification task. The result of the feature extraction stage is called a *feature vector*.

4. **Classification**, in which the feature vector extracted from the object is classified into the most appropriate class.

5. **Post-processing**. As different actions might also have different costs associated with them, the final task of a pattern recognition system is to decide upon an action based on the classification results.

Figure 2.1 illustrates the different stages for an optical character recogniser.

While the sensing, pre-processing and feature extraction tasks are very specific to the particular problem, the classification task can be somehow generalised for typical pattern recognition systems. Formally, a classifier can be defined as a function:

$$c(x) = \arg\max_{c \in \mathcal{C}} g_c(x) \tag{2.1}$$

where, for each class c, a *discriminant function* g_c is used to measure the degree to which the object x belongs to class c. Obviously, x is classified into a class to which it belongs with maximum degree. When random variables are used for x and

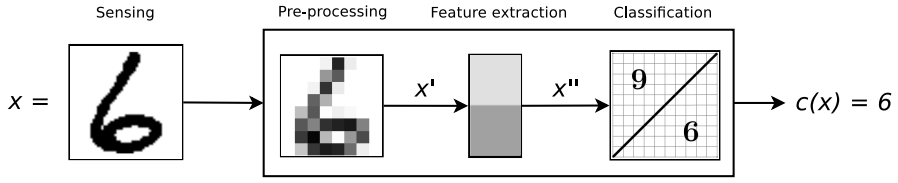

Figure 2.1: Optical character recogniser for digits 6 and 9, based on the average gray levels of the upper and the bottom half.

c, the optimal classification technique is the so-called *Bayes classifier* or *decision rule* (named after Thomas Bayes):

$$c^*(x) = \arg\max_{c \in \mathcal{C}} p(c|x) \tag{2.2}$$

If the posterior $p(c|x)$ probability is modeled directly, the classifier is said to follow a discriminative approach. However, the classical approach to pattern recognition is to write the Bayes classifier in terms of class priors $p(c)$ and class-conditional densities $p(x|c)$ (generative approach). By applying the Bayes' rule:

$$c^*(x) = \arg\max_{c \in \mathcal{C}} p(c|x) = \arg\max_{c \in \mathcal{C}} p(c)p(x|c) \tag{2.3}$$

Labelled samples $(x_1, c_1), ..., (x_N, c_N)$ randomly drawn from $p(x, c)$ are used to estimate $p(c)$ and $p(x|c)$. Usually, for each class c, its prior is estimated as:

$$p(c) \approx \frac{N_c}{N} \qquad [N_c = \sum_{n:c_n=c} 1] \tag{2.4}$$

and its conditional density $p(x|c)$ is estimated from samples labelled with c.

2.1.2 Automatic speech recognition

Automatic speech recognition (ASR) can be defined as the independent, computer-driven transcription of spoken language into readable text. In a nutshell, ASR is technology that allows a computer to identify the words that a person speaks into a microphone and convert it to written text. Having a machine to understand fluently spoken speech has driven speech research for more than 50 years. Although ASR technology is not yet at the point where machines understand all speech, in any acoustic environment, or by any person, it is used on a daily basis in a number of applications and services.

Formally, the speech recognition problem can be described as a function that defines a mapping from the acoustic evidence to a single or a sequence of words. Let $\boldsymbol{x} = (x_1, x_2, ..., x_t)$ represent the acoustic evidence that is generated in time from a given speech signal. Let $\boldsymbol{w} = (w_1, w_2, ..., w_n)$ denote a sequence of n words, each belonging to a fixed set of possible words, $\mathcal{W}$. Let $p(\boldsymbol{w}|\boldsymbol{x})$ denote the probability

that the words $\boldsymbol{w}$ were spoken given that the acoustic evidence $\boldsymbol{x}$ was observed. The speech recogniser should select the sequence of words $\hat{\boldsymbol{w}}$ satisfying:

$$\hat{\boldsymbol{w}} = \arg\max_{w \in \mathcal{W}} p(\boldsymbol{w}|\boldsymbol{x}) \tag{2.5}$$

However, since it is dificult to directly model $p(\boldsymbol{w}|\boldsymbol{x})$, we can apply the Bayes' rule as in Equation 2.3:

$$\hat{\boldsymbol{w}} = \arg\max_{w \in \mathcal{W}} p(\boldsymbol{w}|\boldsymbol{x}) = \arg\max_{w \in \mathcal{W}} p(\boldsymbol{w})p(\boldsymbol{x}|\boldsymbol{w}) \tag{2.6}$$

where $p(\boldsymbol{w})$ is known as *language model* and $p(\boldsymbol{x}|\boldsymbol{w})$ as *acoustic model*. Typically, *n-gram models* are used to estimate $p(\boldsymbol{w})$ [21]; while $p(\boldsymbol{x}|\boldsymbol{w})$ is estimated using *Hidden Markov Models* (HMMs) and *Gaussian Mixture Models* (GMMs) [25]. Figure 2.2 shows a general overview of an automatic speech recognition system.

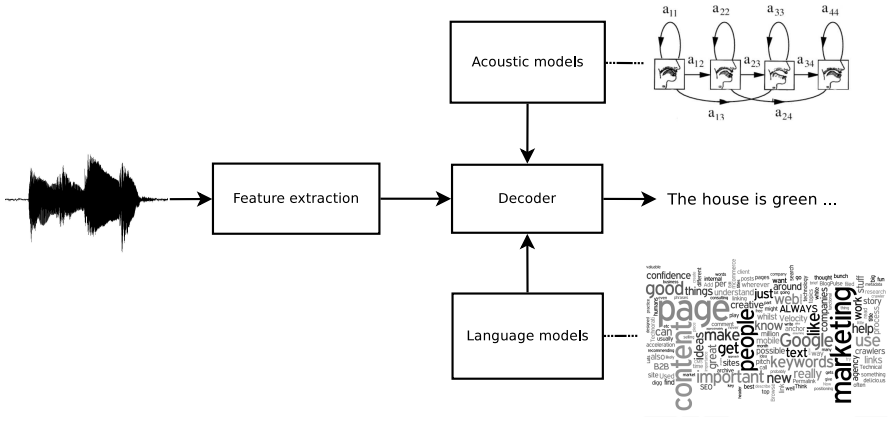

Figure 2.2: General overview of an automatic speech recognition system.

2.1.3 Statistical machine translation

Machine translation (MT) is the use of computers to automate the translation of texts or utterances from one language into another, while the underlying meaning remains the same. The history of MT goes back to the late 40s with the famous publication of Weaver [49], widely recognised as one of the pioneers of machine translation. The 70s and 80s saw the proliferation of rule-based machine translation systems such as Meteo [42], Systran [8] and METAL [6]. The contributions in statistical machine translation were minor until the early 90s, when the IBM group presented the Candide system [7]. Since then, the development of statistical MT has experienced a major boost on account of the many research groups emerged in this area.

The goal of MT is the automatic translation of a source sentence $\boldsymbol{s}$ into a target sentence $\boldsymbol{t}$, being

$$\begin{array}{ll} \boldsymbol{s} = s_1, ..., s_j, ..., s_{|s|} & s_j \in \mathcal{S} \\ \boldsymbol{t} = t_1, ..., t_i, ..., t_{|t|} & t_i \in \mathcal{T} \end{array}$$

where s_j and t_i denote source and target words, and $\mathcal{S}$ and $\mathcal{T}$, the source and target vocabularies, respectively.

In statistical MT, this translation process is usually presented as a decision process, where given a source sentence $\boldsymbol{s}$, a target sentence $\hat{\boldsymbol{t}}$ is selected according to

$$\hat{\boldsymbol{t}} = \arg\max_t p(t|s) \tag{2.7}$$

where $p(t|s)$ is the probability for t to be the actual translation of s. Applying the Bayes' rule we can reformulate Equation 2.7 as

$$\hat{\boldsymbol{t}} = \arg\max_t p(t)p(s|t) \tag{2.8}$$

where $p(t)$ and $p(s|t)$ correspond to the *language* and *translation* models, respectively. Intuitively, the language model represents the well-formedness of the candidate translation t; while the translation model can be understood as a mapping function from target to source words.

Most of the state-of-the-art statistical MT systems are based on bilingual pharses [9, 23]. Another approach which has become popular in recent years is grounded on the integration of syntactic knowledge into statistical MT systems [51, 52, 15]. The third main approach is the modelling of the translation process as a finite-state transducer [12, 4].

2.2 Video lecture repositories

In this section we present the poliMedia and VideoLectures.NET video lecture repositories. Both of them have been transcribed and translated as part of the transLectures project. transLectures' ASR and MT systems have been specially developed to exploit the particular characteristics that large video lecture repositories usually present. More specifically, the acoustic, language and translation models are adapted to the particular speaker and topic of the lectures, thereby improving the transcriptions and translations accuracy [27, 2, 3]. This adaptation process is referred to as *massive adaptation*.

At this point, it is worth stressing that the integration of transLectures' technologies into these repositories is one of the major contributions of this thesis and will be properly addressed in Chapter 3.

2.2.1 poliMedia

The poliMedia platform is a recent, innovative service for the creation and distribution of educational multimedia content at the UPV [33]. It was initially designed to allow UPV professors to generate and publish educational video lectures. It currently serves to more than 30 000 students and 2800 professors. The poliMedia platform has also

Tables 2.1: Basic statistics of the UPV's poliMedia repository (May 2014)

Number of lectures	11 662
Total duration (in hours)	2422
Avg. lecture length (in minutes)	12.5
Total number of speakers	1443
Avg. number of lectures recorded per speaker	7

been exported to several both national and international universities. Table 2.1 shows basic statistics of the current UPV's poliMedia repository.

The production process of the poliMedia platform has been carefully designed to generate high quality recordings with a high production rate. The poliMedia studio is a 4 meter room with a white background in which all the necessary equipment for the recording is available to professors. Figure 2.3 shows the studio during a recording session.

Figure 2.3: The poliMedia studio during a recording session.

Recordings on poliMedia follow a particular standard format. As it can be seen in Figure 2.4, the speaker appears on the bottom right corner of the screen while the computer screen (usually showing the time-aligned presentation slides) is shown centered as the main element of the recording. The videos are stored in AVC/H.264 format with different settings. Video size is 1280x720 pixels and average bit rates usually oscillate between 500 and 900 kbps. Audio format is AAC/LC stereo (85%) and mono (15%) with sampling frequencies of 44 100 and 48 000 Hz. According to the Nyquist–Shannon sampling theorem, this is more than sufficient to accurately cover the human speech frequency range (∼200-3500 Hz).

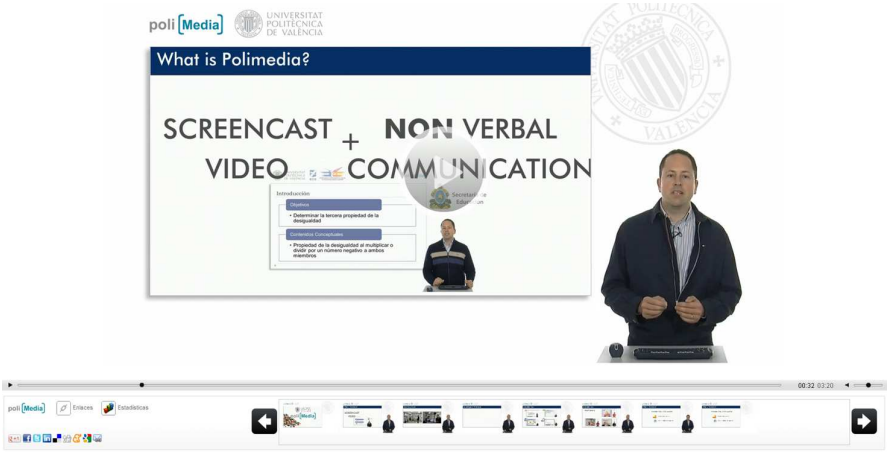

Figure 2.4: A poliMedia recording example.

In order to automatically transcribe the poliMedia Spanish repository, automatic speech recognition systems need relevant sample data. To this end, 114 hours of poliMedia Spanish video lectures were manually transcribed. This data was properly partitioned into training, development and test sets in order to train, tune and evaluate the ASR systems. Details regarding each set are shown in Table 2.2.

Tables 2.2: Statistics of the Spanish poliMedia training, development and test partitions

	Training	Development	Test
Videos	655	26	23
Speakers	73	5	5
Hours	107h	3.8h	3.4h
Sentences	39.2K	1.3K	1.1K
Words	936K	35K	31K
Vocabulary	26.9K	4.7K	4.3K

2.2.2 VideoLectures.NET

VideoLectures.NET is a free and open access repository of video lectures mostly filmed by people from the Jožef Stefan Institute (IJS) at major conferences, summer schools, workshops and science promotional events from many fields of Science. VideoLectures.NET is being used as an educational platform for several EU funded research projects, different open educational resources organizations such as the OpenCourseWare Consortium, MIT OpenCourseWare and Open Yale Courses as well as other

scientific institutions like CERN. In this way, VideoLectures.NET collects high quality educational content which is recorded with high-quality, homogeneous standards.

All lectures, accompanying documents, information and links are systematically selected and classified through the editorial process taking into account the author's comments. The video editing is done in-house and is never censored, that is, lectures are never edited in a way which would allow content or viewer manipulation. Most lectures are accompanied with time-aligned presentation slides (as shown in Figure 2.5).

Tables 2.3: Basic statistics of the IJS' VideoLectures.NET repository (May 2014)

Number of lectures	20 358
Total number of authors	12 167
Total duration (in hours)	12 681
Average lecture duration (in minutes)	37

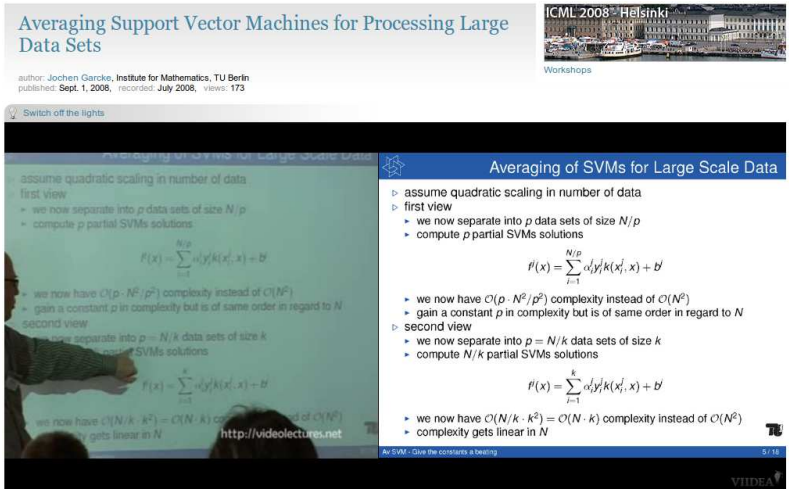

Figure 2.5: The VideoLectures.NET web player.

In order to train proper ASR systems to automatically transcribe the repository, high-quality transcriptions were manually generated for an English subset of VideoLectures.NET. The resulting training, development and test sets are summarised in Table 2.4.

Tables 2.4: Statistics of the English VideoLectures.NET training, development and test partitions

	Training	Development	Test
Videos	20	4	4
Speakers	68	11	25
Hours	20h	3.2h	3.4h
Sentences	5K	1K	1.3K
Words	130K	28K	34K
Vocabulary	7K	3K	3K

2.3 The Opencast Matterhorn platform

Matterhorn is a free, open-source software to support the management of educational audio and video content. Higher education institutions use Matterhorn to produce lecture recordings, manage existing video, serve designated distribution channels, and provide user interfaces to engage students with educational video. The project combines individual solutions and focuses the efforts and experience of different universities in one shared open product. The creation of a unified system with an open development process is projected to foster the exchange of educational content also. To this end, it includes the following features:

- Administrative tools for scheduling automated recordings, manually uploading files, and managing videos, metadata, workflows and processing functions.
- Integration with recording devices in the classroom for managing automated capture of audio, VGA, and multiple video sources.
- Processing and encoding services that prepare and package the media files according to configurable specifications, including rich media features (slide segmentation/indexation) for in-video search.
- Distribution to local streaming and download servers and configuration capability for distribution to channels such as YouTube, iTunes or a campus course or content management system
- Rich media user interface for learners to engage with content, including slide preview, content-based search, heatmaps and additional features.

Multiple educational institutions have adopted Opencast Matterhorn as their video Content Management System (CMS). The University of California Berkeley, the University of Vigo or the University of Osnabrueck are some examples currently involved into the Opencast Community. The international success of Opencast Matterhorn makes the platform a perfect scenario for the integration of state-of-the-art automatic speech recognition and machine translation technologies. By integrating transLectures tools into the Matterhorn platform, educational institutions will be

able to break the language barrier and support people with hearing disabilities by subtitling their videos in multiple languages.

Chapter 3

The transLectures Platform

3.1 Introduction

The transLectures Platform [41] comprises a set of tools for integrating automatic transcription and translation technologies into large educational video repositories. Its main components are the transLectures Database, Web Service, Ingest Service and Player. These tools have been used to integrate transLectures' ASR and MT systems into the previously described poliMedia and VideoLectures.NET repositories [39].

Figure 3.1 gives a general overview of the transLectures Platform integration into poliMedia, VideoLectures.NET or any other video lecture repository. In this figure, the principal connections between the different tools in a client repository are illustrated. The transLectures Database, Web Service and Ingest Service will be described in detail in Sections 3.2, 3.3 and 3.4. The transLectures Player, one of the author's main contributions, is given special attention in Section 3.5.

3.2 transLectures Database

The transLectures Database is a PostgreSQL relational database which stores all the data required for the Web Service and the Ingest Service. The main categories of data stored in the Database are as follows:

- **Video lectures:** All the information related to a specific lecture is stored in the database, including language, duration, title, keywords and category. In addition, an external ID, provided by the client repository, is stored and used for lecture identification purposes in all transactions performed between the client and the Player and Web Service.

- **Speakers:** Information about the lecture speaker can be used by the ASR systems to adapt the underlying models to the unique characteristics of a given speaker and, therefore, improve the quality of the resulting subtitles.

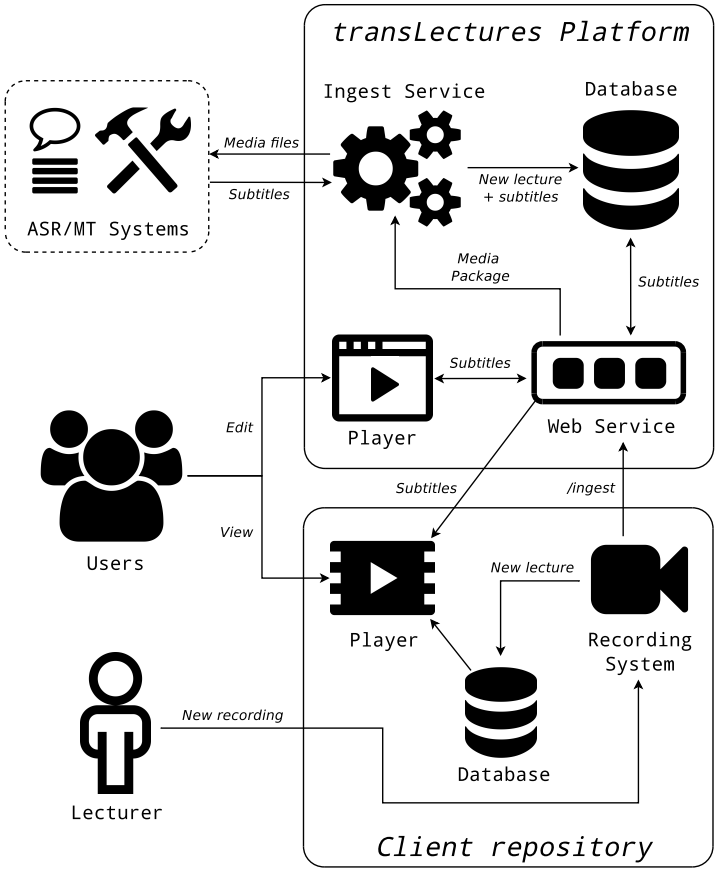

Figure 3.1: General overview of the transLectures Platform integration into a client repository.

- **Subtitles:** All subtitles generated via the Ingest Service are stored in the database and retrieved by the Web Service.
- **Uploads:** Every time an /ingest operation is performed, a new upload entry is stored in the database.

Media and subtitle files are stored on the hard drive separately from the relational database. We can distinguish between three different kind of files:

- **Media:** Video/audio files of the lectures that already exist in the database, plus related files such as slides, external documents and thumbnails.

- **Transcriptions:** Subtitle files in DFXP format.
- **Uploads:** Uploaded Media Package Files (MPF) and the files contained within them.

3.3 transLectures Web Service

The *transLectures* Web Service is the interface for transfering information and data between the poliMedia, VideoLectures.NET or any other repository and the transLectures Platform. It also enables the subtitle visualisation and editing capabilities of the transLectures Player. This web service is implemented as a Python Web Server Gateway Interface (WSGI), and defines a set of HTTP interfaces related to subtitle delivery and media upload:

- */ingest*: POST request which allows the client to upload audio/video files and other related material, such as slides and other text resources that can be used to adapt the ASR system, together with other metadata in a Media Package File (MPF). This MPF will be later processed by the Ingest Service in order to generate automatic transcriptions and translations for the uploaded media.
- */status*: GET request to check the status of a video lecture uploaded via the /ingest interface.
- */lecturedata*: GET request that returns basic metadata and file locations for a given video lecture.
- */langs*: GET request that provides the client with a list of subtitles and languages available for a given video lecture.
- */dfxp*: GET request that returns the subtitles in DFXP format for a given lecture and language.
- */mod*: POST request that sends and commits changes made by a user when editing a transcription or translation. User corrections are later used by ASR and SMT systems in order to improve the underlying models.

3.4 transLectures Ingest Service

The Ingest Service is the tool devoted to handling and properly processing the Media Package Files uploaded via the /ingest interface of the Web Service. It is implemented as a Python module that should be executed periodically (typically every minute) to check for and process new lecture uploads, and also to assess whether existing uploads are being processed correctly. The *uploads table* of the Database is used to keep the status of every upload up-to-date. This information is also accessed by the Web Service's /status interface.

As it was previously mentioned, MPFs are uploaded to the transLectures Platform via the Web Service's /ingest interface and stored in the Database. Then the Ingest Service reads the uploads table of the Database and starts processing each MPF. An upload will typically follow the following sequential steps (see Figure 3.2):

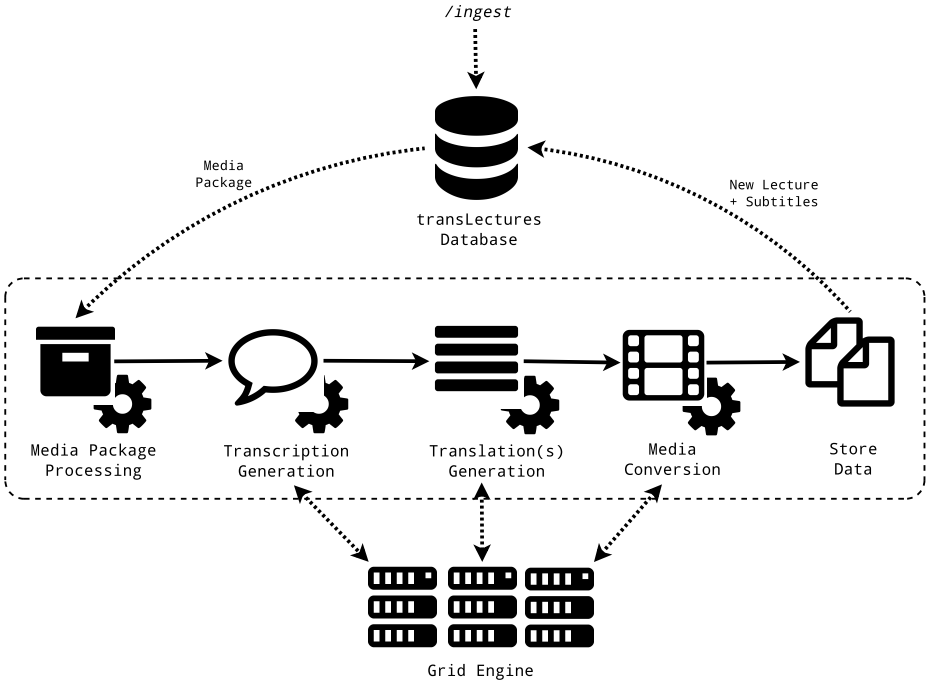

Figure 3.2: Overview of the transLectures Ingest Service workflow.

1. **Media Package Processing:** In this step the MPF is unzipped and a series of security, data integrity and data format checks are performed on the unpacked data. If all checks come up clean, then the MPF data is redirected to the next step, which might be the 2nd, 3rd, 4th or 5th step listed here, depending on the input data.

2. **Transcription Generation:** In this step a transcription file in DFXP format is generated from the main media file (video, audio) using a suitable ASR Module.

3. **Translation(s) Generation:** In this step one or more translation files in DFXP format will be generated from the transcription file (whether automatically generated in the previous step or provided in the MPF) using suitable MT Modules.

4. **Media Conversion:** In this step the main media file is converted into the video formats required by the transLectures Player in order to maximise browser compatibility.

5. **Store Data:** In this last step, the data contained in the MPF and the data automatically generated by the Ingest Service are stored in the Database.

3.5 transLectures Player

Massive adaptation can deliver substantial contributions to the improvement of transcriptions overall quality, but it is our belief that sufficiently accurate results are unlikely to be obtained through fully-automated approaches alone. Instead, in order to reach the desired levels of accuracy, we must consider user interaction. For that purpose, an HTML5 post-editing application has been carefully designed to expedite the error supervision task [46], and thereby obtain subtitles of an acceptable quality in exchange for a minimum amount of user effort. Figure 3.3 illustrates the communication between the Player and the other tools in the platform.

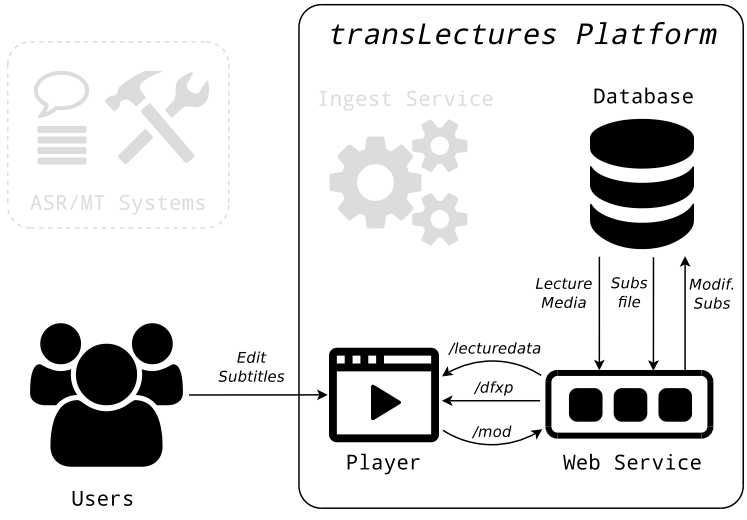

Figure 3.3: transLectures Player communications diagram.

In this section we describe the three different versions of the tool that were developed and evaluated by users of both poliMedia and VideoLectures.NET platforms.

3.5.1 Standard post-editing

In the standard post-editing approach, users are shown automatic transcriptions and translations while viewing the lecture. The standard layout is illustrated in Figure 3.4.

However, three alternative editing layouts are available for users to choose from according to their personal preferences. In any of them, users can freely supervise any transcription and translation segment. Additionally, a complete set of key shortcuts has been implemented to enhance expert user capabilities.

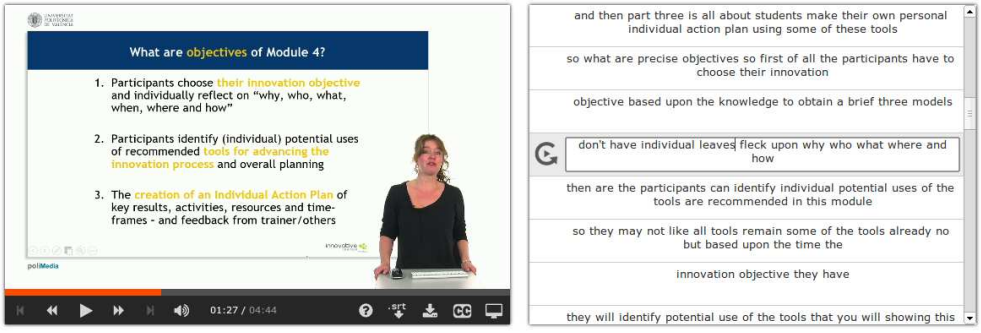

Figure 3.4: transLectures Player standard post-editing mode (default side-by-side layout).

The user interaction can be summarised as follows: the video lecture and the corresponding transcription or translation are played in synchrony, allowing users to read the transcription while watching the video. When a user spots an error, they can click on the particular segment to pause the video and enter their changes.

3.5.2 Intelligent interaction

Standard user models for the transcription of audiovisual objects, like the one presented above, are batch-oriented. These models yield satisfying results when highly collaborative users are working on near-perfect system output. To find out whether we could further improve supervision times, an alternative interaction strategy, based on confidence measures [35], was introduced. These confidence measures provide an indicator as to the probable correctness of each word appearing in the automatic transcription. Words with low confidence values are likely to have been incorrectly recognised at the point of ASR. The idea is that by focusing supervision actions only on incorrectly-transcribed words, we can optimise user interaction to get the best possible transcription in exchange for the least amount of effort [36, 40].

Figure 3.5 shows the intelligent interaction interface. Here, users are asked to supervise a subset of words preselected by the CAT (*computer-assisted translation*) system as low confidence. This subset typically constitutes between 10-20% of all words transcribed using the ASR system, though users are able to modify this range at will to as low as 5% and as high as 40%, depending on the perceived accuracy of the transcription. Each word was played in the context of one word before and one word after, in order to facilitate its comprehension and resulting correction. In the figure, low-confidence words are shown in red and corrected low-confidence words in

Figure 3.5: transLectures Player intelligent interaction interface (only editing box is shown).

green. The text box that opens for each low-confidence word can be expanded in either direction in order to modify the surrounding text as required.

3.5.3 Two-step supervision

Intelligent interaction can quickly improve the transcription accuracy with limited user effort. Nevertheless, as the CAT system will unlikely find all possible transcription errors, a small amount of incorrect words will remain. The system could use the intelligent interaction inputs as constraints in the ASR search space in order to find the best transcription hypothesis. Figure 3.6 gives an overview of the two-step supervision strategy.

Basically, in the first step, low-confidence words are presented to the user for supervision in increasing order of confidence. These words keep being presented until one of the three following conditions are met:

- The total supervision time reaches double the duration of the video itself.
- No corrections are entered for five consecutive segments.
- 20% of all words are supervised.

Then, supervision actions are fed into the ASR system to generate a new and

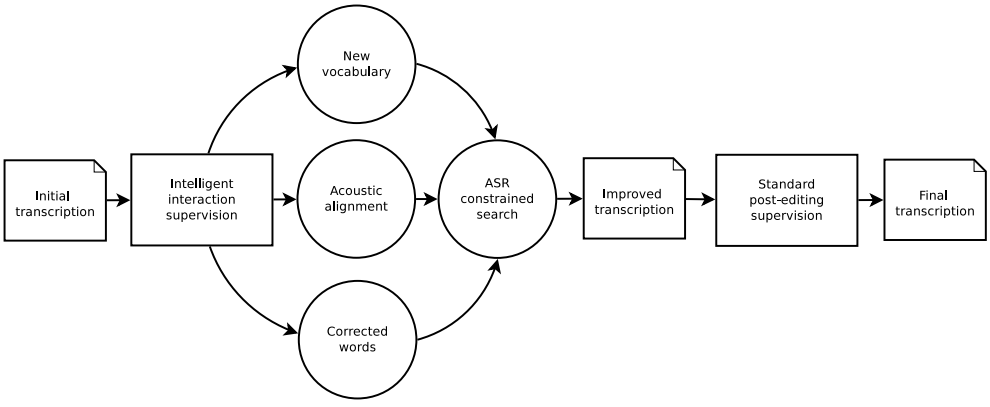

Figure 3.6: transLectures Player two-step supervision strategy overview.

improved transcription. In the second step, users supervise the improved transcription from start to finish, following the standard post-editing method.

3.5.4 User evaluations

User evaluations were carried out within the transLectures project in order to evaluate the models and tools in a real-life setting. UPV lecturers, having already filmed material for poliMedia, were invited under 2012-2013 and 2013-2014 *Docència en Xarxa* calls to evaluate the computer-assisted transcription tools being developed in transLectures. Lecturers signing up for this programme committed to supervising the automatic transcriptions of five of their poliMedia videos using the tools described in this chapter.

In order to evaluate the different interaction strategies, UPV lecturers were asked to rate various aspects on a Likert scale from 1-10 (see Table 3.2). In addition, usage statistics were collected to objectively evaluate the real user effort (see Figure 3.7). The metric used to that end was the *Real Time Factor* (RTF), which is the ratio (R) between the time spent by the user ellaborating the transcription (P) and the duration of the video (T).

$$R = \frac{P}{T} \tag{3.1}$$

Figure 3.7 shows the RTF as function of WER[1] for the different interaction strategies. Table 3.1 shows detailed information regarding this figure, which is discussed below.

[1]WER stands for *word error rate*, and it is based on the Levenshtein distance string metric. Informally, the WER between two sentences is the minimum number of word edits (i.e. insertions, deletions or substitutions) required to change one sentence into the other, divided by the original sentence length.

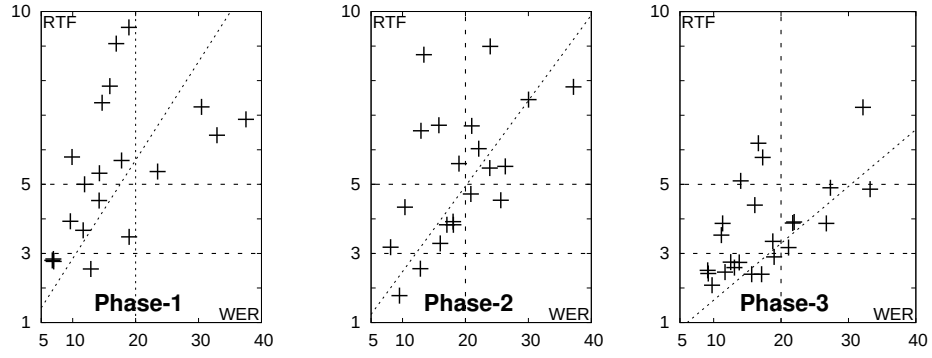

Figure 3.7: RTF as a function of WER for the different interaction strategies. Phases 1, 2 and 3 correspond to *standard post-editing*, *intelligent interaction* and *two-step supervision*, respectively.

1. *Standard post-editing*: The average WER of the initial automatic transcriptions was 16.9, and the average RTF for the post-editing process was 5.4. When compared to that recorded for transcribing from scratch (∼10 RTF for non-expert users [29]), we got a significant decrease of about 50%. This way, lecturers' performance became comparable to that of professional transcriptionists [17], rather than that expected from non-expert transcriptionists [28].

2. *Intelligent interaction*: The mean RTF was lowered to 2.2, but it must be noted that the resulting transcriptions were not error free. The WER was reduced (in average) from 19.5 to 8.0 after the intelligent interaction supervision. These results underline the effectiveness of confidence measures.

3. *Two-step supervision*: The mean RTF for the first step was as low as 1.4. Although it only reduced the average WER from 28.4 to 25.0, the ASR massive adaptation and constrained search post-process was able to lower the WER of the transcriptions to 18.7. Then, the improved transcriptions were supervised following the standard post-editing strategy (RTF 3.9). The cumulative RTF was 5.3. Although this RTF is comparable to that obtained for the standard post-editing strategy, it should be stressed that the initial WER was 28.4 (w.r.t. 16.9) in this case.

3.5.5 Conclusions

According to Table 3.2, we can see that lecturer preferences do not rely on pure efficiency metrics. Although standard post-editing WER reduction per RTF unit was the lowest, UPV lecturers rated it as the best choice. When they were questioned

Tables 3.1: Summary of results obtained for each interaction strategy.

	SP-E (1)	II (2)	T-SS (3)
Initial WER	16.9	19.5	28.4
Final WER	0.0	8.0	0.0
RTF	5.4	2.2	5.3
∇WER/RTF	3.1	5.2	5.4

Tables 3.2: Satisfaction survey results for each interaction strategy.

Question (summarised)	Mean (SP-E)	Mean (II)	Mean (TS-S)
Intuitiveness			
1- Easy to use.	9.4	7.8	7.5
2- Easy to learn.	9.4	8.1	8.6
3- Help information clear.	9.2	8.1	8.5
4- Organisation on screen clear.	9.0	8.4	8.7
Grand Mean	*9.3*	*8.1*	*8.3*
Likeability			
5- Comfortable.	8.7	6.5	7.3
6- Like the interface.	8.7	6.9	7.4
7- I am satisfied.	9.0	6.9	7.4
Grand Mean	*8.8*	*6.8*	*7.4*
Usability			
8- Effectively complete work.	9.0	6.7	7.7
9- Quicker than from scratch.	8.6	6.6	7.4
10- Has everything I expect.	9.0	5.6	7.1
Grand Mean	*8.9*	*6.3*	*7.4*
Overall Mean	*9.1*	*7.2*	*7.8*

about the intelligent interaction and the two-step supervision strategies, they answered they did not want to leave any error in the transcriptions, and they did want to avoid supervising the same lecture twice.

However, their ratings might be influenced by many factors. For instance, it is understandable that users rate better a tool for editing an automatic transcription when the initial WER is 16.9 than when it is 28.4. Another factor that might stress their obsession for leaving no errors in the transcriptions is the fact that they were editing their own lecture transcriptions.

What we can conclude is that the standard post-editing interaction is more intuitive and user-friendly than the intelligent interaction and two-step supervision alternatives, which might require of some expertise. Nevertheless, if a relatively small

amount of time is going to be spent on supervising a transcription, and the accuracy strictness is not tight, intelligent interaction and two-step supervision have proven to be significantly more efficient choices.

Chapter 4

Integration into Opencast Matterhorn

4.1 Introduction

In this chapter we address the integration of transLectures' ASR and MT solutions into the Opencast Matterhorn platform, which was introduced in Section 2.3. The goal is to integrate the tools described in Chapter 3 into the different Matterhorn workflow phases. This is discussed below in three sections: Section 4.2 describes the system architecture of the Matterhorn platform. Next, a more detailed analysis of the platform, focusing on relevant details for integration purposes, is presented in Section 4.3. Finally, the strategies followed for the successfull integration of the tools are described in Section 4.4.

4.2 System architecture

The Matterhorn platform comprises four modules: lecture capture and administration, ingest and processing, distribution and engage tools. Figure 4.1 shows a diagram of the Matterhorn architecture which includes its main components and dependencies among them.

The members of the Opencast Community have selected Java as programming language to create the necessary applications and a Service-Oriented Architecture (SOA) infrastructure. The overall application design is highly modularised and relies on the OSGi (dynamic module system for Java) technology. The OSGi service platform provides a standardised, component-oriented computing environment for cooperating network services.

The different phases of the Matterhorn workflow are applied as follows:

1. **Schedule/prepare and capture**: The recording process begins with determining what is to be recorded, where and in what form. Campus data will be

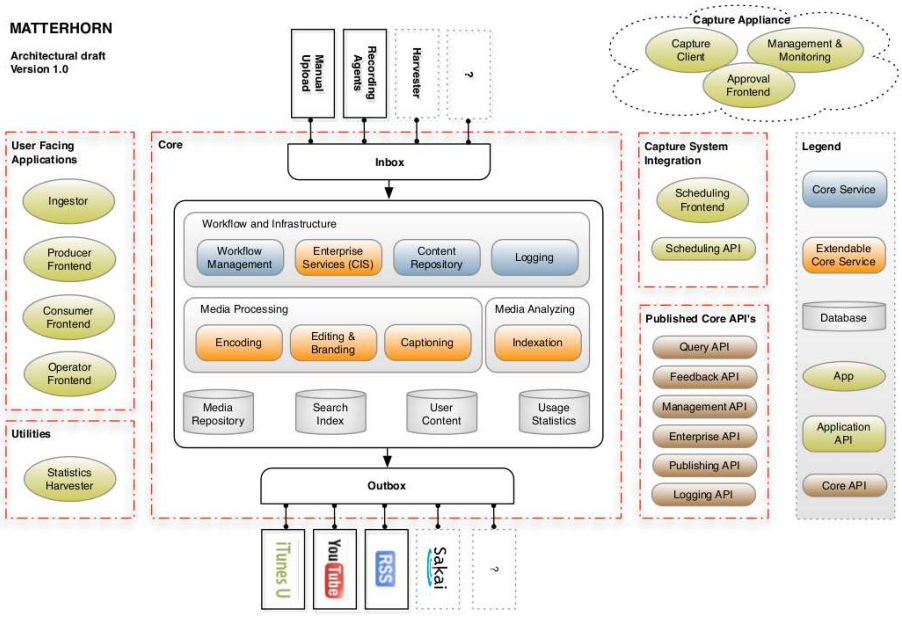

Figure 4.1: Matterhorn architectural draft.

integrated by the universities' IT departments. For this purpose, Matterhorn is open to both the learning management systems and administrative data bases. Syllabi, lecture and room timetables do not only provide the basic information to answer the question raised above, but in an ideal case, most of the metadata related to the recording (lecturer, title, summary, language, etc.) as well. Recording devices are then scheduled to automatically record, e.g. in lecture hall 0S03, every Tuesday from 10:00 to 12:00, the lecture on "XYZ" by Prof. ABC.

2. **Ingest and processing**: At the end of the recording the tracks are sent to an "inbox" to be processed. The inbox also serves as "ingest" for other video objects to be integrated in the subsequent workflows of Matterhorn. The different recording tracks (audio, content, video) are bundled to a media package, content-indexed (at first through optical character recognition of the slide, later certainly through audio recognition also) and if necessary archived in the most native formats. They are encoded according to the specified distribution parameters.

3. **Distribution**: The distribution demands of the universities are extremely heterogeneous: they go from simple integration of the videos in local WCMS or blogs, to posting in password-protected LMS, to distribution via iTunes U or

YouTube. The distribution module uploads ingested content into different distribution channels according to particular institutions needs.

4. **Engage tools**: This module is closely linked to the distribute module since it must also manage presentation and use of the objects. However, applications in the Engage module make it possible to use comprehensive information (metadata, video and audio analysis, annotations, and use analysis) for intelligent user interfaces. Likewise, support of learning management systems (LMS) or virtual learning environments (VLE) is an important issue. To make sure that the produced material will be used, Matterhorn video and audio player components are easily integrated in existing course websites, wikis, and blog systems. Social annotations, which can be used to improve search or navigation and feedback possibilities are flown back to the system like the user statistics already mentioned.

4.3 Platform analysis

In order to successfully integrate the transLectures' tools into the Opencast Matterhorn platform, further analysis of the platform must be done. In this section, we describe in detail different aspects of the Matterhorn platform which are relevant for this purpose.

4.3.1 Media package

After audio/video material has been sent to the inbox, the media is bundled into a media package. A media package is considered the business document within the Matterhorn system. Besides the media objects, it includes further information from media analysis as well as metadata. Every media package therefore consists of a manifest and a list of package elements that are referred to in the manifest. Package elements are:

- Media tracks (audiovisual material)
- Metadata catalogs
- Attachments (slides, pdf, text, annotations, etc.)

The media package generation process is carried out automatically when new material is uploaded to the system (see Figure 4.2).

4.3.2 Workflow

Once a media package has been successfully generated, it goes through a set of operations, like the encoding of the media files in different formats or the publication of the media in distribution channels. These operations are defined in a *workflow*. A Matterhorn workflow is an ordered list of operations defined in a XML file. There is no

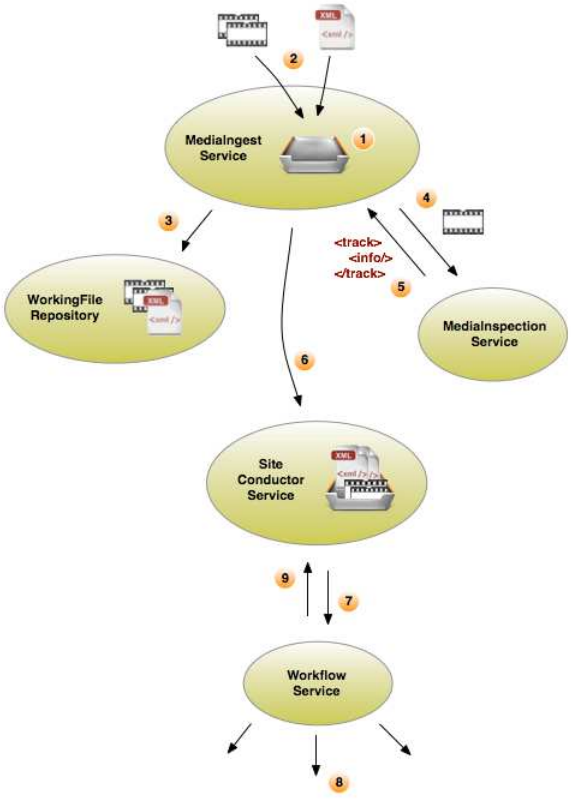

Figure 4.2: Media package ingest process.

limit to the number of operations or their repetition in a given workflow. A workflow operation can run autonomously or pause itself to allow for external, usually user, interaction. Although there are some predefined workflows in the default Opencast Matterhorn installation, custom workflows can be defined in order to perform the desired operations to all media packages being ingested into the system.

Workflow operations are usually calls to different platform services. These operations are defined in the Matterhorn's Conductor Service and are known as *workflow operation handlers*. Some basic workflow operations included in Matterhorn are, for instance, the *inspect* operation which extracts technical information about the media files included in the media package, the *compose* operation for encoding video and audio files in different formats, or the *archive* operation to store the files in the system. Custom workflow operation handlers can be implemented in order to extend workflow functionalities.

4.4 Integration process

The integration of transLectures' tools into the Matterhorn platform can be divided in two parts. The first part (Section 4.4.1) will cover the generation of automatic transcriptions and translations for the ingested media. The second part (Section 4.4.2) will involve the visualisation and supervision of the automatically generated captions. The overall integration process is illustrated in Figure 4.3.

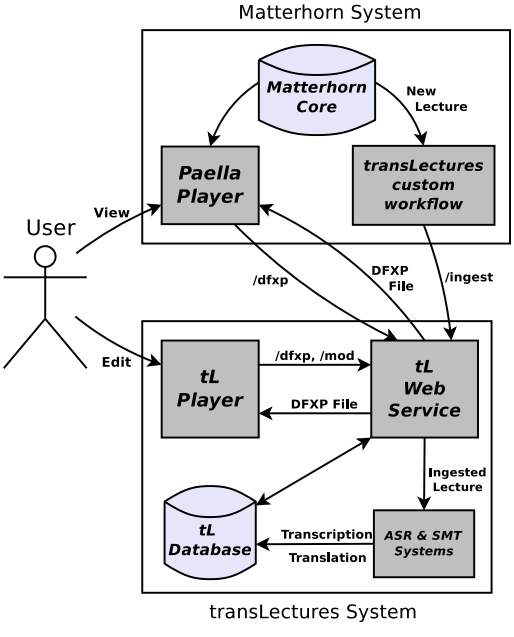

Figure 4.3: Matterhorn integration overview.

4.4.1 Generation of automatic transcriptions and translations

Figure 4.2 showed how ingested media is sent to the Workflow Service in order to be processed by a particular workflow. As mentioned in Section 4.3.2, custom workflows can be defined to perform specific operations during the ingest process. In order to send the media files to the transLectures Platform, a custom workflow including a custom workflow operation handler will be defined. In this way, audio tracks extracted from the ingested media files will be wrapped up in a transLectures media package and sent to the transLectures Web Service *ingest* interface. To summarise:

1. Media tracks are ingested into the Matterhorn platform.

2. The transLectures workflow processes the media tracks:

(a) Initial standard Matterhorn operations are performed (*inspect*, *prepare-av*, *compose*, *trim*, etc.)

(b) Audio tracks are extracted from media files (FLAC format)

(c) A transLectures media package is generated, including:

- Extracted audio files
- Media language
- Matterhorn mediaPackage ID
- Title
- Author
- Duration
- Attachments (slides, documents, etc.)

(d) The media package is sent to the transLectures Web Service */ingest* interface.

(e) Final standard Matterhorn operations are performed (*segmentpreviews*, *publish*, etc.)

The described process is illustrated in Figure 4.4.

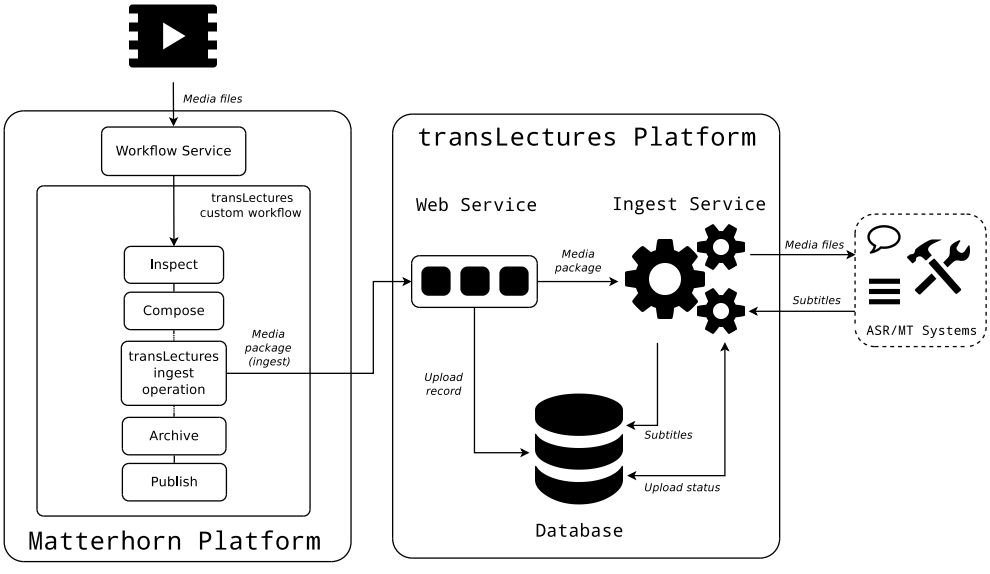

Figure 4.4: transLectures custom workflow overview.

4.4.2 Subtitle visualisation and supervision

The generated subtitles must be made available to the users of the platform when accessing the media. Furthermore, as it was meant in transLectures, users should be

able to supervise the automatic transcriptions and translations in order to improve its accuracy. To this purpose, an alternative to the Matterhorn Engage Player is presented.

The Paella Player[1], developed by the *Área de Sistemas de Información y Comunicaciones* (ASIC) at the UPV, is an HTML5 multistream video player capable of playing multiple audio and video streams synchronously and supporting a number of user plugins. It is specially designed for lecture recordings and fully compatible with the Matterhorn platform.

Paella Player also provides support for the transLectures Platform tools. It is capable of displaying transLectures subtitles by directly accessing the transLectures Web Service. In addition, it also provides a link to supervise the transcriptions and translations by using the transLectures Player. Figure 4.5 illustrates the subtitle visualisation support of Paella Player.

Figure 4.5: Paella Player showing available subtitles for a Matterhorn lecture.

[1] Visit http://paellaplayer.upv.es for more information.

Chapter 5

Using speech transcriptions in lecture recommender systems

5.1 Introduction

One problem created by the success of video lecture repositories is the difficulty faced by individual users when choosing the most suitable video for their learning needs from among the vast numbers available on a given site. Users are often overwhelmed by the amount of lectures available and may not have the time or knowledge to find the most suitable videos for their learning requirements. Up until recently, recommender systems have mainly been applied in areas such as music [24, 30], movies [11, 50], books [31] and e-commerce [13], leaving video lectures largely to one side. Only a few contributions to this particular area can be found in the literature, most of them focused on VideoLectures.NET [5]. However, none of them has explored the possibility of using lecture transcriptions to better represent lecture contents at a semantic level.

In this chapter we describe a content-based lecture recommender system that uses automatic speech transcriptions, alongside lecture slides and other relevant external documents, to generate semantic lecture and user models. In Section 5.2 we give an overview of this system, focusing on the text extraction and information retrieval process, topic and user modeling and the recommendation process. In Section 5.3 we address the dynamic update of the recommender system and the required optimisations needed to maximise the scalability of the system. The integration of the system presented in Sections 5.2 and 5.3 into VideoLectures.NET, carried out as part of the PASCAL Harvest Project La Vie, is described in detail in Section 5.4. Finally, we close with some concluding remarks, in Section 5.5.

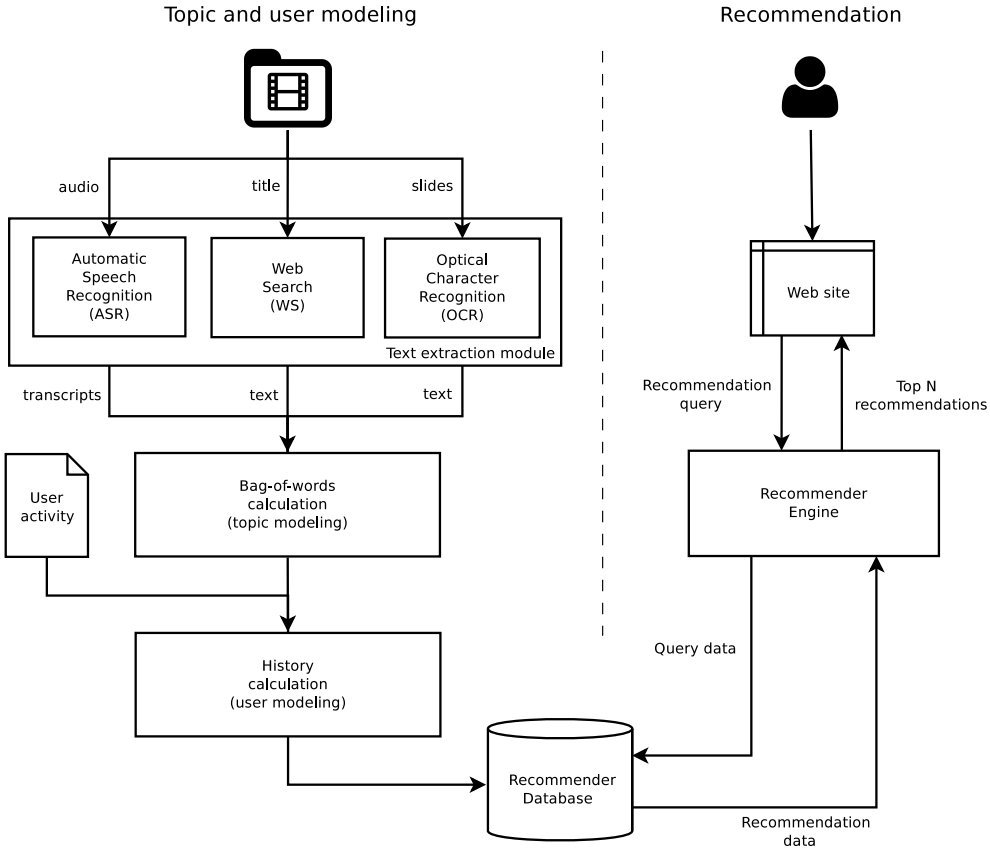

Figure 5.1: System overview.

5.2 System overview

Fig. 5.1 gives an overview of the recommender system. The left-hand side of the figure show the topic and user modeling procedure, which can be seen as the training process of the recommender system. To the right we see the recommendation process. The aim of topic and user modeling is to obtain a simplified representation of each video lecture and user. The resulting representations are stored in a recommender database. This database will be exploited later in the recommendation process in order to recommend lectures to users.

As shown in Fig. 5.1, every lecture in the repository goes through the topic and user modeling process, which involves three steps. The first step is carried out by the text extraction module. This module comprises three submodules: ASR (Automatic Speech Recognition), WS (Web Search) and OCR (Optical Character Recognition).

As its name suggests, the ASR submodule generates an automatic speech transcription of the video lecture. The WS submodule uses the lecture title to search for related documents and publications on the web. The OCR submodule extracts text from the lecture slides, where available. The second step takes the text retrieved by the text extraction module and computes a *bag-of-words* representation. This bag-of-words representation consists of a simplified text description commonly used in natural language processing and information retrieval. More precisely, the bag-of-words representation of a given text is its vector of word counts over a fixed vocabulary. Finally, in the third step, lecture bags-of-words are used to represent the users of the system. That is, each user is represented as the bag-of-words computed over all the lectures the user has ever seen.

When the topic and user modeling process ends, the recommender database is ready for exploitation by the recommender engine (see the right-hand side of Fig. 5.1). This engine uses recommendation features to calculate a measure s of the suitability of the recommendation for every (u, v, r) triplet, where u refers to a particular user, v is the lecture they are currently viewing and r is a hypothetical lecture recommendation. In recommender systems, this is usually referred to as the *utility function* [34]. Specifically, it indicates how likely it is that a user u would want to watch lecture r after viewing lecture v. For instance, this utility function can be computed as a linear combination of recommendation features:

$$s(u, v, r) = \boldsymbol{w} \cdot \boldsymbol{x} = \sum_{n=1}^{N} w_n \cdot x_n \tag{5.1}$$

where $\boldsymbol{x}$ is a feature vector computed for the triplet (u, v, r), $\boldsymbol{w}$ is a feature weight vector and N is the number of recommendation features. In this work, the following recommendation features were considered:

1. *Lecture popularity:* number of visits to lecture r.

2. *Content similarity:* weighted dot product between the lecture bags-of-words v and r [19].

3. *Category similarity:* number of categories (from a predefined set) that v and r have in common.

4. *User content similarity:* weighted dot product between the bags-of-words u and r.

5. *User category similarity:* number of categories in common between lecture r and all the categories of lectures the user u has watched in the past.

6. *Co-visits:* number of times lectures v and r have been seen in the same browsing session.

7. *User similarity:* number of different users that have seen both v and r.

Feature weights $\boldsymbol{w}$ can be learned by training different statistical classification models, such as support vector machines (SVMs), using positive and negative (u, v, r) recommendation samples.

The most suitable recommendation $\hat{r}$ for a given u and v is computed as follows:

$$\hat{r} = \arg\max_{r} s(u, v, r) \tag{5.2}$$

However, in recommender systems the most common practice is to provide the user the M recommendations r that achieve the highest utility values s, for instance, the first 10 lectures.

5.3 System updates and optimisation

Lecture repositories are rarely static. They may grow to include new lectures, or have outdated videos removed. Also, users' learning progress or interactions with the repository influence the user models. The recommender database must therefore be constantly updated in order to include the new lectures added to the repository and update the user models. Furthermore, the addition of new lectures to the system might lead to changes to the bag-of-words (fixed) vocabulary. Any variation to this vocabulary involves a complete regeneration of the recommender database. That said, changes to the vocabulary may not be significant until a substantial percentage of new lectures has been added to the repository.

Two different update scenarios can be defined: the incorporation of new lectures and updating the user models, on the one hand, and the redefiniton of the bag-of-words vocabulary, including the regeneration of both the lecture and user bags-of-words, on the other. We will refer to these scenarios as *regular update* and *occasional update*, respectively, after the different periodicities with which they are meant to be run.

- *Regular update*: The regular update is responsible for including the new lectures added to the repository and updating the user models with the last user activity, both in the recommender database. As its name suggests, this process is meant to be run on a daily basis, depending on the frequency with which new lectures are added to the repository, since new lectures cannot be recommended until they have been processed and included in the recommender database.

- *Occasional update*: As mentioned in Section 5.2, lecture bags-of-words are calculated under a fixed vocabulary. Since there is no vocabulary restriction on the text extraction process, we need to modify the bag-of-words vocabulary as new lectures are added to the system. The occasional update carries out the process of updating this vocabulary, which involves recalculating both the lecture and user bags-of-words.

In order to maximise the scalability of the system, while also reducing the response time of the recommender, the features *Content similarity*, *Category similarity*, *Co-visits* and *User similarity* described in Section 5.2 are precomputed for every possible

lecture pair and stored in the recommender database. Then, during the recommendation process, the recommender engine loads the values of these features, leaving the computation of features *User content similarity* and *User category similarity* until runtime. The decision to calculate the features *User content similarity* and *User category similarity* at runtime was driven by the highly dynamic nature of the user models, in contrast to the lecture models, which remain constant until the bag-of-words vocabulary is changed.

5.4 Integration into VideoLectures.NET

The proposed recommendation system was implemented and integrated into the VideoLectures.NET repository during the PASCAL2 Harvest Project *La Vie* (*Learning Adapted Video Information Enhancer*) [32]. Said integration is discussed here across three subsections. First, in Section 5.4.1 we address the generation of lecture and user models from video lecture transcriptions and other text resources. Next, in Section 5.4.2 we describe how recommender feature weights were learned from data collected from the existing VideoLectures.NET recommender system. Finally, we present our evaluation of the system in Section 5.4.3.

5.4.1 Topic and user modeling

The first step in generating lecture and user models involved collecting textual information from different sources. In particular, for VideoLectures.NET, the text extraction module gathered textual information from the following sources:

- transLectures speech transcriptions.
- Web search-based textual information from Wikipedia, DBLP and Google (abstracts and/or articles).
- Text extracted from lecture presentation slides (PPT, PDF or PNG using Optical Character Recognition (OCR)).
- VideoLectures.NET internal database metadata.

Next, the text extraction module output was used to generate lecture bags-of-words for every lecture in the repository. These bags-of-words, as mentioned in Section 5.2, were calculated under a fixed vocabulary that was obtained by applying a threshold to the number of different lectures in which a word must appear in order to be included. By means of this threshold, vocabulary size is significantly reduced, since uncommon and/or very specific words are disregarded. Once defined, term weights were calculated using *term frequency-inverse document frequency* (td-idf), a statistical weighting scheme commonly used in information retrieval and text mining [26]. Specifically, tf-idf weights are used to calculate the features *Content similarity* and *User content similarity*. Finally, the VideoLectures.NET user activity log was parsed in order to obtain values for the feature *Co-visits* for all possible lecture pairs, as

well as a list of lectures viewed per user. This list was used together with the lectures bags-of-words to generate the users bags-of-words and categories. These, in turn, were used to calculate *User content similarity* and *User category similarity*, respectively, as well as *User similarity* for all possible lecture pairs. In a final step, all this data was stored in the recommender database in order to be exploited by the recommender engine in the recommendation process.

5.4.2 Learning recommendation feature weights

Once the data needed to compute recommendation feature values for every possible (u, v, r) triplet in the repository was made available, the next step was to learn the optimum feature weights $\boldsymbol{w}$ for the calculation of the utility function shown in Equation 5.1. To this end, an SVM classifier was trained using data collected from the existing VideoLectures.NET naïve recommender system (based only on keywords extracted from the lecture titles). Specifically, every time a user clicked on any of the 10 recommendation links provided by this recommender system, 1 positive and 9 negative samples were registered. SVM training was performed using the SVM^{light} open-source software [20]. The optimum feature weights were those that obtained the minimum classification error over the recommendation data.

5.4.3 Evaluation

Although there are many different approaches to the evaluation of recommender systems [37, 18], it is difficult to state any firm conclusions regarding the quality of the recommendations made until they are deployed in a real-life setting. The La Vie project therefore provided an ideal evaluation framework, being deployed across the official VideoLectures.NET site. The strategy followed for the objective evaluation of the La Vie recommender was to compare it against the existing VideoLectures.NET recommender by means of a *coin-flipping* approach. Specifically, this approach consisted of logging user clicks on recommendation links provided by both systems on a 50/50 basis and comparing the total number of clicks recorded for each system.

The results did not show any significant differences between the two recommenders in terms of user behaviour. This can be explained by the fact that user-click count alone is not a legitimate point of comparison for recommendation quality. For instance, random variables not taken into account might influence how users respond to the recommendation links provided. As an alternative, we can compare the rank of the recommendations clicked by users within each system. Specifically, for each recommendation clicked by a user in either system, we can compare how the same recommendation ranked in the other system. This might be a more appropriate measure for comparing the recommendations in terms of suitability. However, additional data need to be collected in order to carry out this alternative evaluation. This data is currently being collected and future evaluation results will be obtained following this rank comparison approach.

Despite the lack of objective evidence for assessing the comparative performance of the La Vie system, subjective evaluations indicate that the proposed recommender

system provides better recommendations than the existing VideoLectures.NET recommender. Fig. 5.2 shows recommendation examples from both systems for a new user viewing a random VideoLectures.NET lecture. Although recommendation suitability is a subjective measure, La Vie recommendations seem to be more appropriate in terms of content similarity.

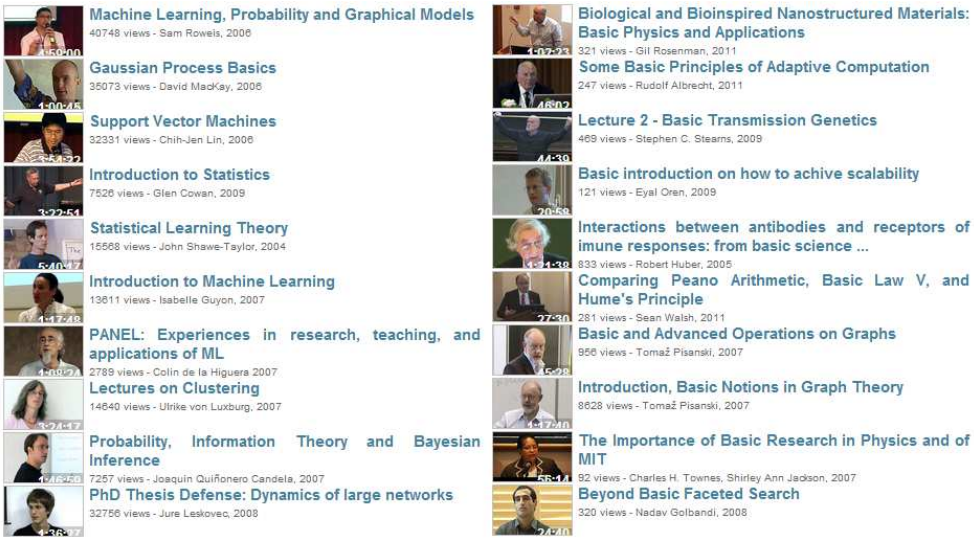

Figure 5.2: On the left, La Vie system recommendations for the "Basics of probability and statistics" VideoLectures.NET lecture. On the right, recommendations made by VideoLectures.NET's existing system for the same lecture.

5.5 Conclusions and future work

In this chapter we have shown how automatic speech transcriptions of video lectures can be exploited to develop a lecture recommender system that can zoom in on user interests at a semantic level. In addition, we have described how the proposed recommender system has been particularly implemented for the VideoLectures.NET repository. This implementation was later deployed in the official VideoLectures.NET site.

The proposed system could also be extended for deployment across more general video repositories, provided that video contents are well represented in the data obtained by the text extraction module.

By way of future work we intend to evaluate the recommender system using other evaluation approaches that measure the suitability of the recommendations more accurately, such as the aforementioned recommendation rank comparison.

Chapter 6

Conclusions and Future Work

6.1 General Conclusions

In this thesis we have presented the integration of state-of-the-art ASR and MT systems, developed within the transLectures project, into different video lecture repositories. In particular, we have shown how transLectures' solutions to produce cost-effective accurate transcriptions and translations have been integrated in poliMedia and VideoLectures.NET (Chapter 3), and also into the Opencast Matterhorn open-source platform (Chapter 4). In addition, in Chapter 5 we have shown how the produced automatic transcriptions can be used to develop a content-based video lecture recommender system.

6.2 Contributions

The scientific publications related to this work are listed below.

- **transLectures.** Silvestre-Cerdà, Joan Albert; Del Agua, Miguel; Garcés, Gonçal; Gascó, Guillem; Giménez-Pastor, Adrià; Martínez, Adrià; Pérez González de Martos, Alejandro; Sánchez, Isaías; Martínez-Santos, Nicolás Serrano; Spencer, Rachel; Valor Miró, Juan Daniel; Andrés-Ferrer, Jesús; Civera, Jorge; Sanchís, Alberto; Juan, Alfons. Proceedings of IberSPEECH 2012, pp. 345-351, 2012.

- **Integrating a state-of-the-art ASR system into the Opencast Matterhorn platform.** Juan Daniel Valor Miró, Alejandro Pérez González de Martos, Jorge Civera and Alfons Juan. IberSPEECH 2012, vol. CCIS 328, Springer, p. 237–246, November 2012, Madrid (Spain).

- **A System Architecture to Support Cost-Effective Transcription and Translation of Large Video Lecture Repositories.** Silvestre-Cerdà, Joan

Albert; Pérez, Alejandro; Jiménez, Manuel; Turró, Carlos; Juan, Alfons; Civera, Jorge. IEEE International Conference on Systems, Man, and Cybernetics (SMC) 2013 , pp. 3994-3999, 2013.

- **Evaluating intelligent interfaces for post-editing automatic transcriptions of online video lectures.** J.D. Valor Miró, R.N. Spencer, A. Pérez González de Martos, G. Garcés Díaz-Munío, C. Turró, J. Civera and A. Juan. Open Learning: The Journal of Open, Distance and e-Learning. Vol. 29, Iss. 1, 2014.

- **Evaluación del proceso de revisión de transcripciones automáticas para vídeos poliMedia.** Valor Miró, Juan Daniel; Spencer, R N; de Martos, Pérez González A; Díaz-Munío, Garcés G; Turró, C; Civera, J; Juan, A. Proc. of I Jornadas de Innovación Educativa y Docencia en Red (IN-RED 2014), Valencia (Spain), 2014.

- **Using Automatic Speech Transcriptions in Lecture Recommendation Systems.** Pérez-González-de-Martos, Alejandro; Silvestre-Cerdà, Joan Albert; Rihtar, Matjaž; Juan, Alfons; Civera, Jorge. IberSPEECH 2014. Submitted.

6.3 Future work

It is our intention to keep improving the different CAT interaction strategies presented in Section 3.5 in order to reduce the required user effort up to a minimum. This might involve to carry out additional user evaluations for gathering more user feedback.

In addition, we will keep updating the transLectures Platform tools presented in Chapter 3 to meet the future requirements of poliMedia, VideoLectures.NET and other video lecture repositories.

Bibliography

[1] Coursera. https://www.coursera.org.

[2] transLectures: First report on massive adaptation. https://www.translectures.eu/wp-content/uploads/2013/05/transLectures-D3.1.1-18Nov2012.pdf.

[3] transLectures: Second report on massive adaptation. http://www.translectures.eu/wp-content/uploads/2014/01/transLectures-D3.1.2-15Nov2013.pdf.

[4] Hiyan Alshawi, Shona Douglas, and Srinivas Bangalore. Learning dependency translation models as collections of finite-state head transducers. *Computational Linguistics*, 26(1):45–60, 2000.

[5] Nino Antulov-Fantulin, Matko Bošnjak, Martin Znidaršic, and Miha et al. Grcar. Ecml-pkdd 2011 discovery challenge overview. *Discovery Challenge*, 2011.

[6] Winfield S. Bennett and Jonathan Slocum. The lrc machine translation system. *Comput. Linguist.*, 11(2-3):111–121, 1985.

[7] Adam L Berger, Peter F Brown, Stephen A Della Pietra, Vincent J Della Pietra, John R Gillett, John D Lafferty, Robert L Mercer, Harry Printz, and Luboš Ureš. The candide system for machine translation. In *Proceedings of the workshop on Human Language Technology*, pages 157–162. Association for Computational Linguistics, 1994.

[8] Reinhard Billmeier. Zu den linguistischen grundlagen von systran. *Multilingua-Journal of Cross-Cultural and Interlanguage Communication*, 1(2):83–96, 1982.

[9] Chris Callison-Burch, Cameron Fordyce, Philipp Koehn, Christof Monz, and Josh Schroeder. Further meta-evaluation of machine translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 70–106. Association for Computational Linguistics, 2008.

[10] Jaime G Carbonell, Ryszard S Michalski, and Tom M Mitchell. An overview of machine learning. In *Machine learning*, pages 3–23. Springer, 1983.

[11] Walter Carrer-Neto, María Luisa Hernández-Alcaraz, Rafael Valencia-García, and Francisco García-Sánchez. Social knowledge-based recommender system. application to the movies domain. *Expert Systems with Applications*, 39(12):10990–11000, 2012.

[12] Francisco Casacuberta and Enrique Vidal. Machine translation with inferred stochastic finite-state transducers. *Computational Linguistics*, 30(2):205–225, 2004.

[13] Jose Jesus Castro-Schez, Raul Miguel, David Vallejo, and Lorenzo Manuel López-López. A highly adaptive recommender system based on fuzzy logic for b2c e-commerce portals. *Expert Systems with Applications*, 38(3):2441–2454, 2011.

[14] Atsushi Fujii, Katunobu Itou, and Tetsuya Ishikawa. Lodem: A system for on-demand video lectures. *Speech Communication*, 48(5):516 – 531, 2006.

[15] Jonathan Graehl and Kevin Knight. Training tree transducers. Technical report, DTIC Document, 2004.

[16] Charles Miller Grinstead and James Laurie Snell. *Introduction to probability*. American Mathematical Soc., 1998.

[17] Timothy J. Hazen. Automatic alignment and error correction of human generated transcripts for long speech recordings. In *Proceedings of the Ninth International Conference on Spoken Language Processing (Interspeech 2006 — ICSLP)*, 2006.

[18] Jonathan L Herlocker, Joseph A Konstan, Loren G Terveen, and John T Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)*, 22(1):5–53, 2004.

[19] Thorsten Joachims. A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. Technical report, DTIC Document, 1996.

[20] Thorsten Joachims. Svmlight: Support vector machine. *SVM-Light Support Vector Machine http://svmlight. joachims. org/, University of Dortmund*, 19(4), 1999.

[21] Slava Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 35(3):400–401, 1987.

[22] Markus Ketterl, Olaf A Schulte, and Adam Hochman. Opencast matterhorn: A community-driven open source software project for producing, managing, and distributing academic video. *Interactive Technology and Smart Education*, 7(3):168–180, 2010.

[23] Philipp Koehn, Franz Josef Och, and Daniel Marcu. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 48–54. Association for Computational Linguistics, 2003.

[24] Seok Kee Lee, Yoon Ho Cho, and Soung Hie Kim. Collaborative filtering with ordinal scale-based implicit ratings for mobile music recommendations. *Information Sciences*, 180(11):2142–2155, 2010.

[25] Stephen E Levinson, Lawrence R Rabiner, and Man Mohan Sondhi. An introduction to the application of the theory of probabilistic functions of a markov process to automatic speech recognition. *Bell System Technical Journal, The*, 62(4):1035–1074, 1983.

[26] Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge, 2008.

[27] Adrià Martínez-Villaronga, Miguel A. del Agua, Jesús Andrés-Ferrer, and Alfons Juan. Language model adaptation for video lectures transcription. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8450–8454. IEEE, 2013.

[28] Cosmin Munteanu, Ron Baecker, and Gerald Penn. Collaborative editing for improved usefulness and usability of transcript-enhanced webcasts. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 373–382, New York, NY, USA, 2008. ACM.

[29] Cosmin Munteanu, Gerald Penn, and Xiaodan Zhu. Improving automatic speech recognition for lectures through transformation-based rules learned from minimal data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 764–772. Association for Computational Linguistics, 2009.

[30] Alexandros Nanopoulos, Dimitrios Rafailidis, Panagiotis Symeonidis, and Yannis Manolopoulos. Musicbox: Personalized music recommendation based on cubic analysis of social tags. *Audio, Speech, and Language Processing, IEEE Transactions on*, 18(2):407–412, 2010.

[31] Edward Rolando Núñez-Valdéz, Juan Manuel Cueva Lovelle, Oscar Sanjuán Martínez, Vicente García-Díaz, Patricia Ordoñez de Pablos, and Carlos Enrique Montenegro Marín. Implicit feedback techniques on recommender systems applied to electronic books. *Computers in Human Behavior*, 28(4):1186–1193, 2012.

[32] PASCAL Harvest Programme. `http://www.pascal-network.org/?q=node/19`.

[33] poliMedia. The polimedia repository, 2007.

[34] Paul Resnick and Hal R Varian. Recommender systems. *Communications of the ACM*, 40(3):56–58, 1997.

[35] Alberto Sanchis, Alfons Juan, and Enrique Vidal. A word-based naïve bayes classifier for confidence estimation in speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(2):565–574, 2012.

[36] Nicolás Serrano, Adrià Giménez, Jorge Civera, Alberto Sanchis, and Alfons Juan. Interactive handwriting recognition with limited user effort. *International Journal on Document Analysis and Recognition*, pages 1–13, 2013.

[37] Guy Shani and Asela Gunawardana. Evaluating recommendation systems. In *Recommender systems handbook*, pages 257–297. Springer, 2011.

[38] Joan Albert Silvestre, Miguel del Agua, Gonçal Garcés, Guillem Gascó, Adrià Giménez-Pastor, Adrià Martínez, Alejandro Pérez González de Martos, Isaías Sánchez, Nicolás Serrano Martínez-Santos, Rachel Spencer, Juan Daniel Valor Miró, Jesús Andrés-Ferrer, Jorge Civera, Alberto Sanchís, and Alfons Juan. translectures. In *Proceedings of IberSPEECH 2012*, 2012.

[39] Joan Albert Silvestre-Cerdà, Alejandro Pérez, Manuel Jiménez, Carlos Turró, Alfons Juan, and Jorge Civera. A system architecture to support cost-effective transcription and translation of large video lecture repositories. In *IEEE International Conference on Systems, Man, and Cybernetics (SMC) 2013*, pages 3994–3999, 2013.

[40] Isaías Sánchez-Cortina, Nicolás Serrano, Alberto Sanchis, and Alfons Juan. A prototype for interactive speech transcription balancing error and supervision effort. In *Proc. of the 2012 ACM Intl. Conf. on Intelligent User Interfaces*, pages 325–326, 2012.

[41] The transLectures-UPV Team. The transLectures Platform (TLP). `http://translectures.eu/tlp`.

[42] Benoît Thouin. The meteo system. *Practical experience of machine translation*, 39:44, 1982.

[43] Jussi Tohka. Sgn-2506: Introduction to pattern recognition. 2006.

[44] The transLectures UPV Team. The translectures-upv toolkit (tlk), 2013.

[45] UPVLC and XEROX and JSI-K4A and RWTH and EML and DDS. translectures. https://translectures.eu/, 2012.

[46] Juan Daniel Valor Miró, Alejandro Pérez González de Martos, Jorge Civera, and Alfons Juan. Integrating a state-of-the-art asr system into the opencast matterhorn platform. In *Advances in Speech and Language Technologies for Iberian Languages*, pages 237–246. Springer, 2012.

[47] Videolectures.NET: Exchange ideas and share knowledge. `http://www.videolectures.net/`.

[48] Mike Wald. Creating accessible educational multimedia through editing automatic speech recognition captioning in real time. *Interactive Technology and Smart Education*, 3(2):131–141, 2006.

[49] Warren Weaver. Translation. *Machine translation of languages*, 14:15–23, 1955.

[50] Pinata Winoto and Tiffany Y Tang. The role of user mood in movie recommendations. *Expert Systems with Applications*, 37(8):6086–6092, 2010.

[51] Dekai Wu. A polynomial-time algorithm for statistical machine translation. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, pages 152–158. Association for Computational Linguistics, 1996.

[52] Kenji Yamada and Kevin Knight. A syntax-based statistical translation model. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 523–530. Association for Computational Linguistics, 2001.

List of Figures

Index of Tables