MeMAD Deliverable

D2.1 Libraries and tools for multimodal content analysis

Grant agreement number	780069
Action acronym	MeMAD
Action title	Methods for Managing Audiovisual Data: Combining Automatic Efficiency with Human Accuracy
Funding scheme	H2020–ICT–2016–2017/H2020–ICT–2017–1
Version date of the Annex I against which the assessment will be made	3.10.2017
Start date of the project	1.1.2018
Due date of the deliverable	31.12.2018
Actual date of submission	31.12.2018
Lead beneficiary for the deliverable	Aalto University
Dissemination level of the deliverable	Public

Action coordinator's scientific representative
Prof. Mikko Kurimo
AALTO–KORKEAKOULUSÄÄTIÖ, Aalto University School of Electrical Engineering, Department of Signal Processing and Acoustics
mikko.kurimo@aalto.fi

MeMAD project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 780069. This document has been produced by theMeMAD project. The content in this document represents the views of the authors, and the European Commission has no liability in respect of the content.

MeMAD

Methods for Managing Audiovisual Data

memad.eu
info@memad.eu

Twitter – @memadproject
Linkedin – MeMAD Project

Authors in alphabetical order		
Name	Beneficiary	e-mail
David Doukhan	INA	ddoukhan@ina.fr
Danny Francis	EURECOM	danny.francis@eurecom.fr
Benoit Huet	EURECOM	benoit.huet@eurecom.fr
Sami Keronen	Lingsoft	sami.keronen@lingsoft.fi
Mikko Kurimo	AALTO	mikko.kurimo@aalto.fi
Jorma Laaksonen	AALTO	jorma.laaksonen@aalto.fi
Tiina Lindh-Knuutila	LLS	tiina.lindh-knuutila@lingsoft.fi
Bernard Merialdo	EURECOM	bernard.merialdo@eurecom.fr
Mats Sjöberg	AALTO	mats.sjoberg@aalto.fi
Umut Sulubacak	UH	umut.sulubacak@helsinki.fi
Jörg Tiedemann	UH	jorg.tiedemann@helsinki.fi
Kim Viljanen	YLE	kim.viljanen@yle.fi

Abstract

This deliverable describes a joint collection of libraries and tools for multimodal content analysis created by the MeMAD project partners. These tools have been further improved and developed during the first year of the project. The description of the components is divided into the visual and auditory domain, and these are further subdivided into different themes. As part of this deliverable, the open source components have been gathered into a joint software collection of tools and libraries publicly available on GitHub. Finally, five scientific publications are appended to the report, which describe the technological advances related to these components that has been made so far in the project.

MeMAD

Methods for Managing
Audiovisual Data

memad.eu
info@memad.eu

Twitter – @memadproject
Linkedin – MeMAD Project

Contents

1 Introduction

This deliverable describes a joint collection of libraries and tools for multimodal content analysis from AALTO, EURECOM, INA, Lingsoft, LLS and Limecraft. The majority of the tools reported here have been created before the MeMAD project, but have been further developed during the first year of the project. In addition there are also tools, which have been created specifically for the project. The tools and libraries described in the current document are needed in the continuation of Work Packages 2, 3 and 5 and also in the task T6.2 Prototype implementation.

In the next section, the overall requirements for the deliverable from the MeMAD project's Description of Action are revisited. Then, we have divided the overview into visual and auditory domains, described in Section 3 and Section 4 respectively. Finally, we include a summary of the components in Section 5 and describe the open source collection of tools that forms the deliverable D2.1 in Section 6. At the end of the report, we have included five scientific publications or their drafts that describe the technological advances made in the project.

2 Requirements of the MeMAD project

The aim of MeMAD Work Package WP2 *Automatic multimodal content analysis* is to develop further the tools and libraries that AALTO, EURECOM, INA, Lingsoft, LLS and Limecraft already have for multimodal analysis, description and indexing of audio and video content. These tools include speech recognition, speaker recognition and diarisation as well as visual and audio description techniques in both uni- and multimodal domains. Developing these automatic methods to meet the needs of the MeMAD project is the content of task T2.1 *Development of automatic tools for multimodal content analysis* and comprises the contents of this deliverable.

The software tools described here are needed both in the continuation of the work package and also in task T6.2 *Prototype implementation*. This report accompanies the software components stored in a GitHub repository with brief descriptions and evaluations of their use. The address of the GitHub repository is:

`https://github.com/MeMAD-project/mmca`

The means of communication between the software components are specified in Deliverables D6.1, D6.4 and D6.7 *Specification of the data interchange format*. The software that is used in the online MeMAD prototype demonstrator is or will be integrated in the Limecraft Flow system. The prototype versions will be described in Deliverables D6.2, D6.5 and D6.8 *MeMAD prototype*. All software components should also work in standalone mode.

3 Visual domain

For the description of the visual content of media, several technical components are needed. In this section we report existing tools and libraries for each type of visual analysis. We start in Section 3.1 by describing the multimedia feature extraction methods used in some of our experiments and available for experimenting in the MeMAD project. In a typical video processing pipeline, the first step is to subdivide it into separate shots which can then be used as separate units for further analysis. Two approaches for this are presented in Section 3.2. The next analysis steps have been subdivided into general visual concept detection (Section 3.3), facial and person recognition (Section 3.4) and emotion and genre recognition (Section 3.5). Finally, at a higher semantic level, visual media content can also be distilled into natural language sentences which aim at holistically describing the media content. This is addressed in Section 3.6.

3.1 Visual features

The aim of visual feature extraction is to create statistical descriptors of visual content (e.g., images or videos) that express the semantically important content of the data in a compact or useful way. Table 1 lists features available in AALTO's content-based multimedia analysis and indexing framework PicSOM. For still images we use pre-trained ResNet [6] Convolutional Neural Network (CNN) models, in particular ResNet-101 and 152. For video data we use a 3D CNN ResNet architecture with 152 layers [7]. We also utilise Faster R-CNN [8] for extracting objects and their locations in the visual scene. The objects are categorised according the 80 object categories of the COCO database (see Section 3.3), and we generate an object location map encoding the rough spatial location of these objects ("frA"). Additionally we also create a feature that discards the location data, and encodes only the occurrence of each object type ("frB"). All the above features are based on using pre-trained models for the Caffe C++ library [9].

For still images, we also use a scene-type cue as a feature based on CNN features extracted for the SUN397 database (see Section 3.3). Scene-type classifiers are designed using Radial Basis Function Support Vector Machine (RBF-SVM) [10], giving a score for each of the 397 scene types.

All image features can be used also for video contents by applying them for the middlemost frame of the clip or shot. As a genuine video feature, the PicSOM system offers dense trajectories [11]. The dense trajectories feature consists of a histogram of oriented gradients (HOG), a histogram of optical flow (HOF), and motion boundary histograms of x and y directions (MBHx and MBHy), which are encoded into a fixed-size vector with a vector quantisation scheme.

Table 1 additionally mentions audio features which encode the occurrence probabilities

abbr.	feature	dim.	modality
rn	ResNet	4096	image
frA	Faster R-CNN	480	image
frB	Faster R-CNN	80	image
s	SUN397	397	image
t	trajectory	5000	video
as	audioset	527	audio

Table 1: Summary of the features used in our experiments.

for the 527 classes of the Google AudioSet Ontology [12]. Our approach, inspired by the works of [13] and [14], uses a multi-level attention model.

3.2 Shot boundary detection

Shot boundary detection aims to split a video into its constituent shots. A shot is a single uninterrupted camera take of a scene. With modern implementations, shot boundary detection can be said to work quite well in all normal cases.

A shot boundary detection service has been implemented as part of the Limecraft Flow system, and its output is used as a supporting process for other services on this platform. Such services include, e.g., the alignment of generated subtitles with shot cut boundaries and optimised user interfaces that display clip timelines by limiting representative images to a single one per pair of shot boundaries. The algorithm is implemented in the C language and is based on algorithms presented in [15].

For using shot boundary detection outside of the Limecraft Flow environment, the Aalto University's content-based indexing framework PicSOM has functionality for shot boundary detection. In PicSOM, the shot boundary detection is based on detecting locally maximal changes in visual features extracted from the video frames. The features typically used are ResNet-50 activations on the pool5 layer [6] and the change of the feature values is evaluated between video frames that are 20 frames apart in time. This normally corresponds to 0.67 seconds, which has been found to be a suitable compromise between getting a good recall of shot boundaries and tolerating gradual shot transitions. The feature-wise differences are thresholded with a preset value to find only the most significant visual changes. Finally, non-maximal suppression is applied in a window of 25 frames (typically 0.83 seconds) to ensure uniqueness of shot boundaries even in moments when the visual scenes have many strong transitions within a short time interval.

3.3 Visual concept detection

Visual concept detection entails labelling media content according to a fixed ontology of concepts, such as certain objects or actions being present in the visual content. Below we list a few central detection categories and the corresponding available datasets and their properties.

COCO object recognition

The Microsoft Common Objects in COntext (COCO) dataset [16] has 2,500,000 labelled object instances in 328,000 images belonging to 80 object categories. COCO is focused on non-iconic views (or non-canonical perspectives) of objects, contextual reasoning between objects, and precise 2D localisation of objects. Examples of COCO concept classes include: *bottle, sofa, chair, motorbike, car, train, cow, dog*, and *person*.

LSCOM concept recognition

The Large-Scale Concept Ontology for Multimedia (LSCOM) [17] defines a vocabulary for annotation and retrieval of video, and has been utilised for example in the TRECVID evaluations [18]. LSCOM includes more than 2000 concepts, which have been selected by multimedia researchers and ontology specialists. There exists also a smaller set of 39 concepts called LSCOM-Lite [19]. Around 400 of the LSCOM concepts, and all of the LSCOM-Lite concepts have been annotated in the TRECVID 2005 development set [20], which consists of 80 hours of video split into almost 62,000 shots. Many subsets of LSCOM have also been annotated in later TRECVID instances, for example 346 concepts were annotated in a collaborative annotation effort [21] in 2011. Examples of LSCOM classes include: *sports, outdoor, mountain, road, crowd, computer/TV screen, people marching*, and *explosion/fire*.

Scene recognition

The SUN Scene Categorization Benchmark database contains 899 scene categories and 130,519 images.[1] Out of these categories, 397 have 100 or more annotated images and these form the SUN397 scene category set [22]. In total SUN397 comprises of 108,754 images, and the categories are divided into a three-level hierarchy. The first level of the hierarchy divides the categories into *indoor, outdoor natural*, and *outdoor man-made*. Examples of category division with all three levels include:
indoor $\rightarrow$ *home or hotel* $\rightarrow$ *alcove*
outdoor natural $\rightarrow$ *water, ice, snow* $\rightarrow$ *bayou*

[1] https://vision.princeton.edu/projects/2010/SUN/

outdoor man-made → *transport* → *airfield*.

Another scene recognition dataset is the MIT Places Database [23] which contains 434 scene categories and 10 million categorised images. The categories are inherited from the original SUN dataset, with some minor changes. Images were downloaded using online web search and category labels verified with crowdsourcing.

Action recognition

The Stanford 40 Action Dataset [24] contains still images of people performing 40 different actions. The dataset contains 9532 images with 180–300 images per action category. Each image also has an associated bounding box around the person performing the annotated action. Examples of action categories include: *riding a horse, rowing a boat, watching TV,* and *using a computer*.

The Google Kinetics-600 dataset [25] contains around 500,000 10-second YouTube video clips annotated with 600 human action categories. Each action has at least 600 associated video clips. The action categories include human-object interaction such as *playing an instrument* and human-human interactions such as *shaking hand* or *hugging*.

dataset	# classes	dataset size	modality	content types
COCO	80	123,287	images	people with objects
LSCOM	350	546,530	images	people, objects, actions, news topics
SUN397	397	100,000	images	scenes, locations
MIT Places	434	10,000,000	images	scenes, locations
Stanford 40	40	10,000	images	actions
Kinetics-600	600	500,000	videos	actions
audioset	527	2,000,000	audio	audio events

Table 2: A summary table of the used visual and aural concept detection methods.

3.4 Facial person recognition

People are undoubtedly an important cue when watching a video. Knowing who appears in a video, when and where, can also lead to learn interesting patterns of relationships among characters of a movie or a news program. In the context of MeMAD, we are interested in generating spatial annotation. Such person-related annotations could provide ground for value added content. Figure 1 shows an example of successful face recognition in broadcast video stream.

There has been much progress in the last decade regarding the process of automatic recognition of individuals. Actually, detection of faces is the first step of the process and it is required before recognition can be performed. The Viola-Jones algorithm [26] for face detection and Local Binary Pattern (LBP) features [27] for the clustering and recognition were the most famous until the advent of deep learning and convolutional neural networks.

Nowadays, two main approaches are in use for detecting faces in video; both are using CNNs. One available within the Dlib library [28] provides good performance for frontal images but requires an additional alignment step (which can also be performed using the Dlib library) before face recognition can be performed when dealing with unconstrained face recognition (which is an essential requirement in generic video analysis scenarios). The recent Multi-task Cascaded Convolutional Networks (MTCNN) [29] approach provides even better performance using an image-pyramid approach and integrates the detection of face landmarks in order to re-align detected faces to the frontal position. For the purpose of MeMAD, the MTCNN approach will be used for locating faces in video frames and for identifying their orientation (thanks to the facial landmarks).

Having located the position and orientation of the faces in the video images, the recognition process can be performed in good conditions. There are several strategies available in the literature to achieve recognition. The basic strategy consists in building a database of faces for each of the persons to recognise and training a classifier to perform the prediction. This approach often suffers from the fact that the addition of new persons requires to retrain the entire model. Currently, the most practical approach is to perform face comparison using a transformation space in which similar faces are close together, and to use this representation to identify individuals. Such embeddings, computed on a large collection of faces are available to the research community. In the context of MeMAD, two such facial representations are being used: Openface [30] and Facenet [31]. Both embeddings are leveraged for jointly predicting the identity of detected persons in video content. This implementation based on multiple off the shelf components will be made available on the projet's GitHub page https://github.com/MeMAD-project/EUR-FaceRec as soon as possible.

3.5 Emotion and genre recognition

Emotion recognition is an active research topic in the affective computing community. During the last decade, emotion recognition systems have been integrated in a number of applications across a growing number of domain fields such as cognitive science [32], clinical diagnosis [33], entertainment [34], and human-machine interaction [35]. Automatic emotion analysis and recognition in real-world videos (*i.e.* in the wild) is nevertheless still an open challenge in computer vision. One fundamental limiting factor is that there is almost no large dataset with real-world facial expressions available for emotion recognition. Other challenging factors include head pose variation, complex facial expression variations, different illumination conditions and face occlusion.

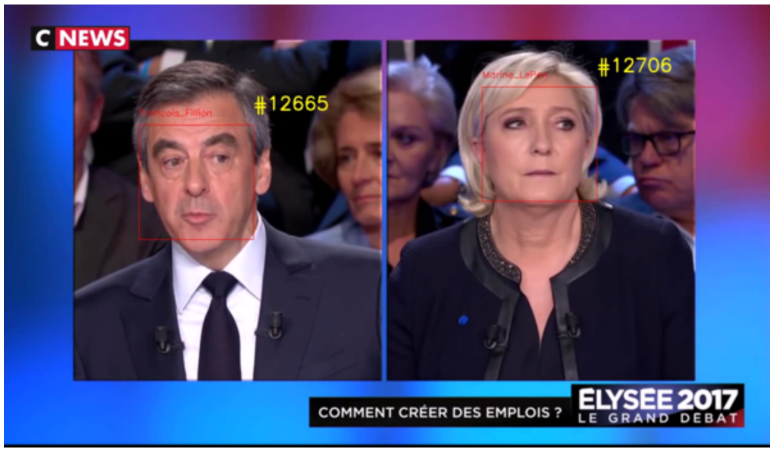


Figure 1: An example of face recognition for broadcast video material

Recent achievements in the field are based on the use of data coming from multiple modalities, such as facial and vocal expressions. Indeed, each modality presents very distinct properties and combining them helps to learn useful and complementary representations of the data. Still, representing and fusing different modalities in an appropriate and efficient manner is an open research question.

The extraction of visual cues for emotion recognition has received a great deal of attention in the past decade. Recently, with the rapid growth of Convolutional Neural Networks, extracting visual features from video frames has been investigated in many emotion recognition tasks and there are various pre-trained face models made available [31, 36, 37]. However, those models are not directly suitable for video due to the lack of temporal information and to the variation of emotion expression patterns across individuals. To deal with this issue, 3D versions of CNN have been recently proposed [38].

Adding the audio information surely plays an important role in emotion recognition in video. Most of the multimodal approaches mainly used hand-crafted audio features such as the Mel Frequency Cepstrum Coefficients (MFCC) or spectrograms, with either traditional [39, 40] or deep [41] classifiers. However, those audio features are very low level (contain little or no semantic information) and are not designed for video analysis.

In the context of MeMAD, we will use a deep multimodal architecture for emotion recognition where both visual and temporal information are represented using a hybrid 2D-3D CNN architecture, while the audio information is extracted using a deep CNN that has been trained by transferring knowledge from vision to sound [42].

Genre classification or identification from media provides important and insightful cues about multimedia items, which can prove to be pivotal to further processing. As a result, there has been significant effort within the text, audio, image and video communities to

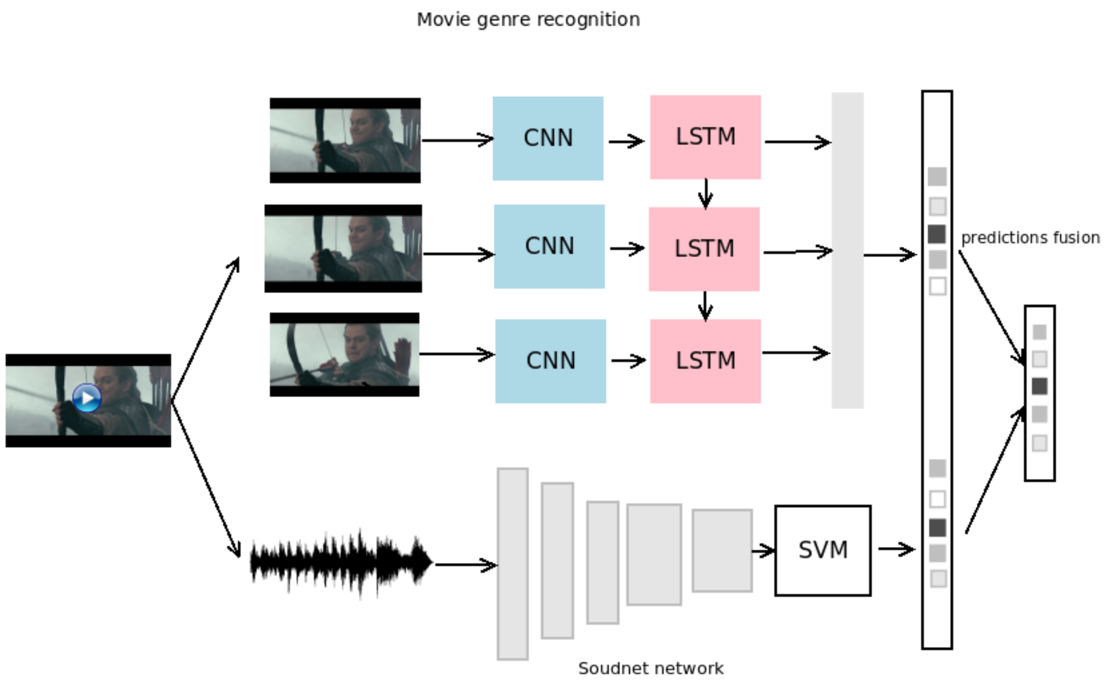


Figure 2: Multimodal deep genre prediction framework

extract such information automatically from the document content [43, 44, 45, 46].

EURECOM has, in the past few years addressed the issue of movie genre classification as part of a higher-level objective aimed at predicting media interestingness [47, 48]. We developed the deep framework illustrated in Figure 2 which include CNN-based visual feature extraction followed by Long Short-Term Memory (LSTM) [49] -based temporal dynamics modelling, as well as deep audio features extraction and learning. We trained the separate genre classifiers (*i.e.* one based on visual and one based on audio features) on a dataset of movie trailers (extended from the one proposed in [50]) assuming that the majority of shots from a trailer are representative of the genre of the original movie. The independently obtained probability vector outputs for visual and audio data respectively are then averaged in order to obtain a global genre distribution for the video shots.

The results obtained show the very acceptable level of performance of the proposed framework. Overall precision in genre prediction reaches 90%. For a more detailed coverage of the approach and its results we invite interested readers to the following publication [51]. This framework will be employed within MeMAD to provide detailed genre information at different content granularity levels (from entire program to specific shots).

3.6 Image and video description

Image or video description entails automatically generating a short text or caption describing the visual contents using only the image or video as the input. AALTO currently has two different Python-based software systems for caption generation. The older one, *NeuraltalkTheano*, uses the Theano library whereas the newer one, *DeepCaption*, uses the PyTorch library. Both are based on the "Show and tell" approach [52], but augmented with several enhancements.

The Theano-based neural captioning system is described in our recent paper [53], and the source code of the implementation is freely available.[2] The neural architecture adds several novel properties including residual connections between the LSTM [49] layers, and the effective usage of persistent features. Persistent features are given as an additional input to the recurrent model at each step of the caption generation, while in the typical setup features are used only for initialising the LSTM generator.

DeepCaption is also released as open source[3] and will be the new base for our future research in the area. DeepCaption development has been initiated during the first year of MeMAD. The goal is to re-implement NeuraltalkTheano in a PyTorch architecture that is more maintainable in the long run, and thus facilitate faster development and experimentation. Our team reached fourth place in the description generation subtask of the TRECVID VTT 2018 competition, and our systems and results are described in AALTO's TRECVID 2018 notebook paper [2] which is also included in this deliverable as Appendix A.2. The development of our results before and after the TRECVID evalation can be seen in Figure 3 as measured in the BLEU-4 metric. We can notice that the performance of the DeepCaption method is close to the best performance obtained in the evaluation. In the continuation of the MeMAD project we will adopt the techniques used by the best performing teams.

EURECOM participated in the matching subtask of TRECVID VTT 2018. In that subtask, participants are given videos and corresponding sentences, and their goal is to match videos to sentences. Results are evaluated in terms of Mean Inverse Rank.

The models we designed for that task are inspired by the Word2VisualVec++ [54] approach, with some enhancements. First, we augmented visual features usually derived frame by frame with global features using an RGB-I3D [55], with a view to take into account not only still objects but also actions. Second, in Word2VisualVec++, text processing is done using a simple Gated Recurrent Unit (GRU) [56]. We tried to replace the GRU by some other models. In particular, we tested the Gated Recurrent Capsules (GRCs) that we introduced in a recent paper [57]. GRCs are an extension of GRUs, which have been designed to attend to important words of a sentence.

[2]https://github.com/aalto-cbir/neuraltalkTheano
[3]https://github.com/aalto-cbir/DeepCaption

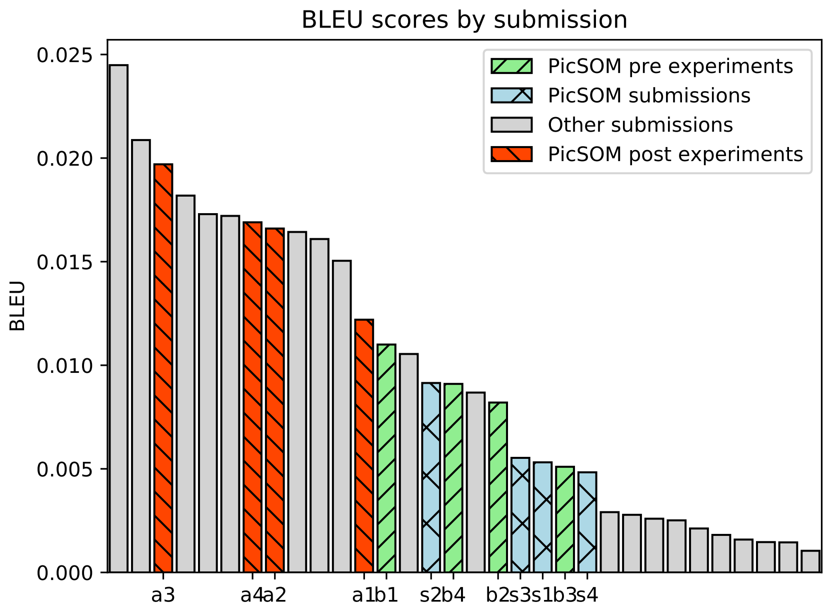


Figure 3: The performance of the PicSOM team in TRECVID VTT 2018 task (blue bars) and its development after the evaluation (red bars). Higher BLEU-4 score is better.

TRECVID VTT 2018 provided us with five datasets of sentences corresponding to a dataset of videos. EURECOM ranked third in the matching subtask of TRECVID VTT 2018 for each of these five datasets. Our models and results are presented with more details in EURECOM's notebook paper [1], which is included in this deliverable as Appendix A.1.

4 Auditory domain

For the description of the audible content of media, several technological tools have been made available. In general the analysis can be divided into detection and recognition of speech, speaker identification and recognition of other audio events. In speech recognition the efforts are focused on accurate automatic transcription of relatively good quality broadcast speech that has large vocabulary and multiple genres, which is considered most essential for the project. The speaker identification and diarisation provides useful information for both the video segmentation and speech transcripts. The audio segmentation can detect active parts and separate speech and music zones. The general audio event classification assigns several potentially overlapping tags to the signal to augment the visual features in the video description.

4.1 Speech recognition

Lingsoft will provide the consortium with transcripts of test material from Yle data set for gold standard evaluation of automatic speech recognition (ASR) and diarisation both in Finnish and Swedish. The Swedish material is ready and shared with the consortium, while the Finnish test material still needs a revision. Lingsoft provides its ASR in Finnish and Swedish for the use in the consortium via a speech service API.

The ongoing development of Finnish ASR includes the accommodation of state of the art neural network based acoustic modelling, improvements in the position dependency of phonemes, and recurrent neural network language modelling (RNNLM) [58] to rescore the first-pass decoded lattices.

The Swedish ASR has been improved by utilising thousands of hours of new audio data extracted from the recordings of Swedish Parliament (Riksdagen) sessions. The data cannot be shared by the consortium, but it is freely downloadable from the Swedish Parliament website `https://riksdagen.se/`. Transcripts of the Swedish parliament sessions have also been incorporated into the language model and improvements have been done to the pre- and postprocessing stages of Swedish ASR that are responsible for appropriate shrinking and expansion of, e.g., numerals, dates and abbreviations.

The baseline Finnish and Swedish ASR systems have been evaluated against the improved systems with challenging conversational multispeaker television broadcast YLE data. The Finnish evaluation set consists of approx. 4.5 hours and Swedish approx. 5.5 hours of speakerwise presegmented audio. Word error rates (WERs) of the systems are gathered in Table 3. Punctuation and case sensitivity have been ignored in computing the error rates.

Table 3: Word error rates of the Lingsoft ASR system.

Language	**Baseline**	**Improved**	**Improved + RNNLM**
Finnish	31.3	26.1	24.8
Swedish	56.0	43.2	-

We also evaluated the Finnish Lingsoft ASR system of Lingsoft with the same Finnish evaluation set as AALTO in [4] with the baseline system achieving 12.7, improved system 10.8 and improved system with RNNLM 10.0 word error rates.

AALTO has developed a language independent ASR scheme that involves lexicon-free language modelling based on character and other subword units and phoneme-free acoustic modeling based on grapheme subword units. This hybrid HMM-DNN (Hidden Markov Model - Deep Neural Network) system using DNN language models and system combination has been described and evaluated for four languages (Finnish, Swedish, Arabic and English) described in article [4] which is also included in this deliverable as Appendix A.4.

The obtained results are the best published ones for each task.

4.2 Speaker identification and diarisation

The task of speaker identification and diarisation is to divide the recordings into single-speaker segments and recognize the speakers. AALTO has studied a deep learning model for overlapping speaker detection and online speaker diarisation (Tuomas Kaseva, MSc thesis, to be submitted). The system consists of three components. First, the voice activity detector finds speech segments. Second, the speech segments are divided into one second wide overlapping windows and each window is classified either for single speaker or overlapping speakers. Finally, the speaker embeddings are computed for each single speaker window and utilised to recognise the speaker identities and speaker changes. In preliminary experiments the performance of the system was encouraging. The online speaker diarisation is mainly intended to annotate real-time speech recognition output with speaker change information, but it may also become useful in segmenting the videos into moments or as features in the video description system.

Lingsoft has developed a diarisation module for Finnish and Swedish The development of the models and the integration of the module into the speech service API is still under development. The module is based on open source software and performs fast diarisation using a deep neural network that maps variable-length utterances to fixed-dimensional embeddings called x-vectors [59].

4.3 Audio event classification

INA has released an open-source audio segmentation software, which is accessible from GitHub `http://github.com/ina-foss/inaSpeechSegmenter`, and packaged as a pip module `https://pypi.org/project/inaSpeechSegmenter/`. This framework has attracted the interest of a developer community: 10 watches, 64 stars and 14 forks observed on GitHub repository in December 2018.

The software performs a three-step segmentation procedure:

- activity detection based on energy
- segmentation into speech and music zones
- segmentation of speech zones into men and women speech segments

A 20 hours long corpus was constituted and annotated, corresponding to the materials identified as hard for the audio segmentation software realised. These materials correspond to music with singing characteristics close to speech (hip hop, french variety), and speech occurring in very noisy conditions, together with very expressive intonations (sport broadcasts, comics).

This corpus was used for the speech and music detection task of Music Information Retrieval Evaluation eXchange (MIREX 2018) `https://www.music-ir.org/mirex/wiki/2018:Main_Page`. 11 systems were submitted to this evaluation, and an additional corpus was realised to evaluate challenger systems.

INA's system was submitted to this evaluation without being trained nor tuned using the test material: the system was implemented before collecting and annotating the corpus. It achieved the best results for the speech detection task on both corpora [3]. The ongoing analysis of challenger's results is aimed at defining which material is the most challenging for the speech analysis community; and produce systems addressing these hard cases. A detailed technical description of INA's system can be found in Appendix A.3.

Audio event classification refers to recognition tasks involving the assignment of one or several labels, such as 'dog bark' or 'doorbell', to a particular audio signal. AALTO has developed a deep learning model following the Google AudioSet ontology to select tags for each second of audio based on the context window with selected length. The two short demo videos are on YouTube `https://www.youtube.com/watch?v=3ht2aEmn_lk` and `https://www.youtube.com/watch?v=cvR3iA5and8`.

AALTO also participated in the challenge of DCASE 2018 Task 2 `http://dcase.community/challenge2018/`. AALTO's deep learning model and competition entry are explained in greater detail in the associated article that was published in the DCASE workshop proceedings [5] which is also included in this deliverable as Appendix A.5. In the competition the system was not ranked far below the best ones, so it can be considered sufficiently near the state-of-the-art.

The detected tags, time codes and probabilities of the audio events will be used in the project together with visual tags and speech recognition results as inputs to the video description system.

5 Summary table of the software components

Table 4 contains a summary of the software components used in the MeMAD project for multimodal content analysis and available for the project's members. Software components that have proprietary license are available for the MeMAD partners as software or as a service. Those that have been identified to have a liberal licensing scheme, such as MIT or Apache 2, are publicly available as source code in MeMAD's GitHub page as will be described in Section 6.

name	provider	license	code	description
PicSOM	AALTO	Apache 2	C++	multimedia content analysis framework
DeepCaption	AALTO	Apache 2	Python3	image and video captioning
AALTO ASR	AALTO	MIT		speech recognition scripts using Kaldi
AudioTagger	AALTO	Apache 2	Python3	audio event classification
OpenNMT-py	AALTO	MIT	Python3	multi-modal image caption translation for WP4
statistical-tools	AALTO	MIT	Python3	tools for creating dataset statistics for WP5
EUR-FaceRec	EURECOM	Apache 2	Python3	tools for detecting, aligning and recognising faces in video
SpeechSegmenter	INA	MIT	Python3	speech vs music segmentation, speaker gender detection
Flow Shot Cut Detector	Limecraft	proprietary	C	subprogram of broadcast video production system
Lingsoft Speech Service	Lingsoft	proprietary	Python3, C++, JavaScript	automatic speech recognition service via an API

Table 4: Software components of MeMAD related to multimodal content analysis.

6 Organisation and use of the source code repository

The MeMAD project's GitHub area is located at:

https://github.com/MeMAD-project

The liberally licensed software components discussed in this report have been specifically collected for the ease of installation in a repository named `mmca`:

https://github.com/MeMAD-project/mmca

The `mmca` repository currently contains the following software tools and libraries as submodules:

- PicSOM – Aalto University's multimedia analysis framework
- DeepCaption – Aalto University's new image and video captioning program
- Speech recognition training scripts for Finnish – Used in Aalto University's speech recognition system
- Audio event classification program – Used to produce audio event recognitions and features

- Multi-modal image caption translation – Aalto University's caption translation program developed for the needs of MeMAD WP4
- Statistical tools for caption dataset analysis – Aalto University's utilities for analysing image and video captioning datasets for the needs of MeMAD WP5
- EUR-FaceRec – EURECOM's utilities to detect, align and recognise faces in videos
- inaSpeechSegmenter – INA's program for speech segmentation

Some of the modules are physically located outside of the MeMAD GitHub project, but theGit submodule mechanism facilitates their seamless availability from their true locations. All of the software packages can be obtained with a single operation:

```
git clone https://github.com/MeMAD-project/mmca.git --recursive
```

Each of the subdirectories created inside the `mmca` directory contains its own further installation and use instructions. Specifically, each package will have up-to-date instructions for installation and usage in a file called `README.md` in the corresponding directory. The licensing information of each submodule is available in a file named `LICENSE`.

7 References

[1] Danny Francis, Benoit Huet, and Bernard Merialdo. EURECOM participation in TrecVid VTT 2018. In *TRECVID 2018, 22nd International Workshop on Video Retrieval Evaluation, November 13-15, 2018, Gaithersburg, USA*, 11 2018.

[2] Mats Sjöberg, Hamed R. Tavakoli, Zhicun Xu, Héctor Laria Mantecón, and Jorma Laaksonen. PicSOM experiments in TRECVID 2018. In *Proceedings of the TRECVID 2018 Workshop*, Gaithersburg, MD, USA, November 2018.

[3] David Doukhan, Eliott Lechapt, Marc Evrard, and Jean Carrive. INA's MIREX 2018 music and speech detection system. *Music Inform. Retrieval Evaluation eXchange (MIREX)*, 2018.

[4] Peter Smit. State-of-the-art speech recognition across languages with subword grapheme lexicons. *TASLP*, 2018. In review.

[5] Zhicun Xu Peter Smit and Mikko Kurimo. The Aalto system based on fine-tuned audioset features for DCASE2018 task2 - general purpose audio tagging. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, Surrey, UK, November 2018.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. Technical report, Microsoft Research, 2015.

[7] Kensho Hara, Hirokatsu Kataoka Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3D CNNs retrace the history of 2D CNNs and ImageNet? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[8] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, 2015.

[9] Yangqing Jia. Caffe: An open source convolutional architecture for fast feature embedding. `http://caffe.berkeleyvision.org/`, 2013.

[10] Markus Koskela and Jorma Laaksonen. Convolutional network features for scene recognition. In *Proceedings of the 22nd ACM International Conference on Multimedia*, Orlando, FL, USA, November 2014.

[11] Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. Action recognition by dense trajectories. In *Proceedings of the 24th IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3169–3176. IEEE Computer Society, 2011.

[12] Jort F Gemmeke, Daniel PW Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R Channing Moore, Manoj Plakal, and Marvin Ritter. Audio set: An ontology and human-labeled dataset for audio events. In *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*, pages 776–780. IEEE, 2017.

[13] Qiuqiang Kong, Yong Xu, Wenwu Wang, and Mark D Plumbley. Audio set classification with attention model: A probabilistic perspective. *arXiv preprint arXiv:1711.00927*, 2017.

[14] Changsong Yu, Karim Said Barsim, Qiuqiang Kong, and Bin Yang. Multi-level attention model for weakly supervised audio classification. *arXiv preprint arXiv:1803.02353*, 2018.

[15] J. Yuan, H. Wang, L. Xiao, W. Zheng, J. Li, F. Lin, and B. Zhang. A formal study of shot boundary detection. *IEEE Transactions on Circuits and Systems for Video Technology*, 17(2):168–186, Feb 2007.

[16] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common objects in context. In *European Conference on Computer Vision (ECCV)*, 2014.

[17] Milind Naphade, John R. Smith, Jelena Tešić, Shih-Fu Chang, Winston Hsu, Lyndon Kennedy, Alexander Hauptmann, and Jon Curtis. Large-scale concept ontology for multimedia. *IEEE MultiMedia*, 13(3):86–91, July-September 2006.

[18] Alan F. Smeaton, Paul Over, and Wessel Kraaij. High-Level Feature Detection from Video in TRECVid: a 5-Year Retrospective of Achievements. In Ajay Divakaran, editor, *Multimedia Content Analysis, Theory and Applications*, pages 151–174. Springer Verlag, Berlin, 2009.

[19] M. R. Naphade, L. Kennedy, J. R. Kender, S.-F. Chang, J. R. Smith, P. Over, and A. Hauptmann. A light scale concept ontology for multimedia understanding for TRECVID 2005. Technical report, IBM, 2005.

[20] L. Kennedy and A. Hauptmann. LSCOM lexicon definitions and annotations version 1.0. Technical Report #217-2006-3, Columbia University, March 2006. DTO Challenge Workshop on Large Scale Concept Ontology for Multimedia.

[21] Stéphane Ayache and Georges Quénot. Video corpus annotation using active learning. In *Proceedings of the 30th European Conference on Information Retrieval (ECIR)*, pages 187–198. Springer, 2008.

[22] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on*, pages 3485–3492. IEEE, 2010.

[23] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1452–1464, 2018.

[24] Bangpeng Yao, Xiaoye Jiang, Aditya Khosla, Andy Lai Lin, Leonidas Guibas, and Li Fei-Fei. Human action recognition by learning bases of action attributes and parts. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1331–1338. IEEE, 2011.

[25] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

[26] Paul Viola and Michael J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, May 2004.

[27] Timo Ahonen, Abdenour Hadid, and Matti Pietikainen. Face description with local binary patterns: Application to face recognition. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 28(12):2037–2041, 2006.

[28] Davis E. King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10:1755–1758, 2009.

[29] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, 23(10):1499–1503, 2016.

[30] Brandon Amos, Bartosz Ludwiczuk, Mahadev Satyanarayanan, et al. Openface: A general-purpose face recognition library with mobile applications. *CMU School of Computer Science*, 2016.

[31] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[32] Francesco Marchi and Albert Newen. Cognitive penetrability and emotion recognition in human facial expressions. *Frontiers in psychology*, 2015.

[33] A. Hargreaves, O. Mothersill, M. Anderson, S. Lawless, A. Corvin, and G. Donohoe. Detecting facial emotion recognition deficits in schizophrenia using dynamic stimuli of varying intensities. *Neuroscience letters*, 2016.

[34] John R. Smith, Dhiraj Joshi, Benoit Huet, Hsu Winston, and Jozef Cota. Harnessing a.i. for augmenting creativity: Application to movie trailer creation. In *Proceedings of ACM Multimedia*, October 23-27, 2017, Mountain View, CA, USA, 2017.

[35] Emile Barkhof, Leo M.J. de Sonneville, Carin J. Meijer, and Lieuwe de Haan. Specificity of facial emotion recognition impairments in patients with multi-episode schizophrenia. *Schizophrenia Research: Cognition*, 2015.

[36] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, et al. Deep face recognition. In *BMVC*, 2015.

[37] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, volume 4, page 12, 2017.

[38] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning Spatiotemporal Features with 3D Convolutional Networks. In *ICCV*, 2015.

[39] Marco Paleari, Benoit Huet, and Ryad Chellali. Towards multimodal emotion recognition: A new approach. In *Proceedings of the ACM International Conference on Image and Video Retrieval*, CIVR '10, pages 174–181, New York, NY, USA, 2010. ACM.

[40] Shriman Narayan Tiwari, Ngoc QK Duong, Frédéric Lefebvre, Claire-Helène Demarty, Benoit Huet, and Louis Chevallier. Deep features for multimodal emotion classification. Technical report, HAL-Inria, 2016.

[41] Shiqing Zhang, Shiliang Zhang, Tiejun Huang, Wen Gao, and Qi Tian. Learning affective features with a hybrid deep model for audio-visual emotion recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 2017.

[42] Yusuf Aytar, Carl Vondrick, and Antonio Torralba. Soundnet: Learning sound representations from unlabeled video. In *NIPS*, 2016.

[43] Ge Song, Yunming Ye, Xiaolin Du, Xiaohui Huang, and Shifu Bie. Short text classification: A survey. *Journal of Multimedia*, 9(5):635, 2014.

[44] Débora C Corrêa and Francisco Ap Rodrigues. A survey on symbolic data-based music genre classification. *Expert Systems with Applications*, 60:190–210, 2016.

[45] Dharti M Bhoraniya and Tushar V Ratanpara. A survey on video genre classification techniques. In *Intelligent Computing and Control (I2C2), 2017 International Conference on*, pages 1–5. IEEE, 2017.

[46] Gabriel S Simões, Jônatas Wehrmann, Rodrigo C Barros, and Duncan D Ruiz. Movie genre classification with convolutional neural networks. In *Neural Networks (IJCNN), 2016 International Joint Conference on*, pages 259–266. IEEE, 2016.

[47] Claire-Hélène Demarty, Mats Sjöberg, Bogdan Ionescu, Thanh-Toan Do, Michael Gygli, and Ngoc Duong. Mediaeval 2017 predicting media interestingness task. In *MediaEval workshop*, 2017.

[48] Olfa Ben-Ahmed, Jonas Wacker, Alessandro Gaballo, and Benoit Huet. Eurecom@ mediaeval 2017: Media genre inference for predicting media interestingnes. In *the Proceedings of the MediaEval 2017 Workshop, Dublin, Ireland*, 2017.

[49] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997.

[50] K S Sivaraman and Gautam Somappa. Moviescope: Movie trailer classification using deep neural networks. *University of Virginia*, 2016.

[51] Olfa Ben-Ahmed and Benoit Huet. Deep multimodal features for movie genre and interestingness prediction. In *2018 International Conference on Content-Based Multimedia Indexing (CBMI)*, pages 1–6. IEEE, 2018.

[52] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[53] Rakshith Shetty, Hamed R.-Tavakoli, and Jorma Laaksonen. Image and video captioning with augmented neural architectures. *IEEE MultiMedia*, 25(2):34–46, April 2018.

[54] Jianfeng Dong, Xirong Li, and Cees GM Snoek. Predicting visual features from text for image and video caption retrieval. *IEEE Transactions on Multimedia*, 2018.

[55] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 4724–4733. IEEE, 2017.

[56] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

[57] Danny Francis, Benoit Huet, and Bernard Merialdo. Gated recurrent capsules for visual word embeddings. In *MMM 2019, 25th International Conference on MultiMedia Modeling, January 8-11, 2019, Thessaloniki, Greece*, Thessaloniki, GRÈCE, 01 2019.

[58] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černocký, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Proceedings of the 11th Annual Conference of the International Speech Communication Association (INTERSPEECH 2010)*, 2010.

[59] David Snyder, Daniel Garcia-Romero, Gregory Sell, Daniel Povey, and Sanjeev Khudanpur. X-vectors: Robust DNN embeddings for speaker recognition. In *Proceeding of International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

memad.eu
info@memad.eu

Twitter – @memadproject
Linkedin – MeMAD Project

A Appendices

A.1 EURECOM's TRECVID 2018 workshop paper [1]

This paper describes the models that the EURECOM team submitted to TRECVID VTT 2018 (matching subtask) and summarises their results.

EURECOM participation in TrecVid VTT 2018

Danny Francis, Benoit Huet, Bernard Merialdo

October 30, 2018

Abstract

This paper describes the submissions of the EURECOM team to the TrecVid 2018 VTT task. We participated in the Sentence Matching subtask. Our approach is to project both descriptive texts and videos in the same vector space through a deep neural network, and to compare them using a cosine similarity. In particular, we compare several variants of sentence embeddings.

1 Introduction

EURECOM participated in the Sentence Matching subtask of the TrecVid 2018 [1] Video-to-Text (VTT) task for the first time. The approach we chose to follow was to improve the approach of the best team of 2017 [8]. The Sentence Matching subtask of the VTT task requires to link videos and sentences describing these videos. Testing data is composed of 1,000 videos, and five datasets of 1,000 sentences, each sentence corresponding to one video. For each video and for each dataset of sentences, teams are asked to rank sentences from the closest to the most dissimilar. Evaluation is performed on each sentence dataset using the Mean Inverse Rank measure.

2 Our Model

2.1 Definition of our model

As stated in Section 1, our model aims at improving the model of [8]. In [8], video embeddings are derived as follows:

- frames are extracted every 0.5 second for each video;
- features vectors $(v_1, ..., v_n)$ are derived from these frames using the penultimate layer of a ResNet-152 [11];
- these features vectors are then fed sequentially into a GRU [6], whose hidden states $(h_1, ..., h_n)$ are concatenated to corresponding features vectors, to obtain contextualized features vectors $(s_1, ..., s_n) = (v_1 \| h_1, ..., v_n \| h_n)$;
- these contextualized features vectors are combined through a soft attention mechanism to form a vector v, which is actually a weighted sum of $s_1, ..., s_n$;
- this vector v is then projected into a vector space after two fully-connected layers with ReLU activations, where each activation is preceded by a batch normalization.

In our model, the same process is applied for computing video embeddings. However, before feeding v into the two fully-connected layers, we concatenated it with a vector that we derived from the video using the last layer of an RGB-I3D [4]. Moreover, [8] used a ResNet-152 trained on ImageNet [7] whereas we used the ResNet-152 trained on ImageNet and finetuned on MSCOCO [13] proposed by [9].

In [8], text-embeddings are derived as follows:

- three text representations are derived (one using an average of Word2Vec [14] embeddings, a second one is a BoW representation and a third one is derived by taking the last hidden state of a GRU) and concatenated;

- the resulting vector is then fed into two consecutive fully-connected layers following the same process as for videos.

Regarding the text-embeddings part of our model, we tried to replace the GRU by GRCs [10] or a bidirectional GRU. GRCs are extensions of GRUs that we proposed in [10], where we showed that they could improve results of GRUs on multimodal matching tasks. Our model is summarized by Figure 1.

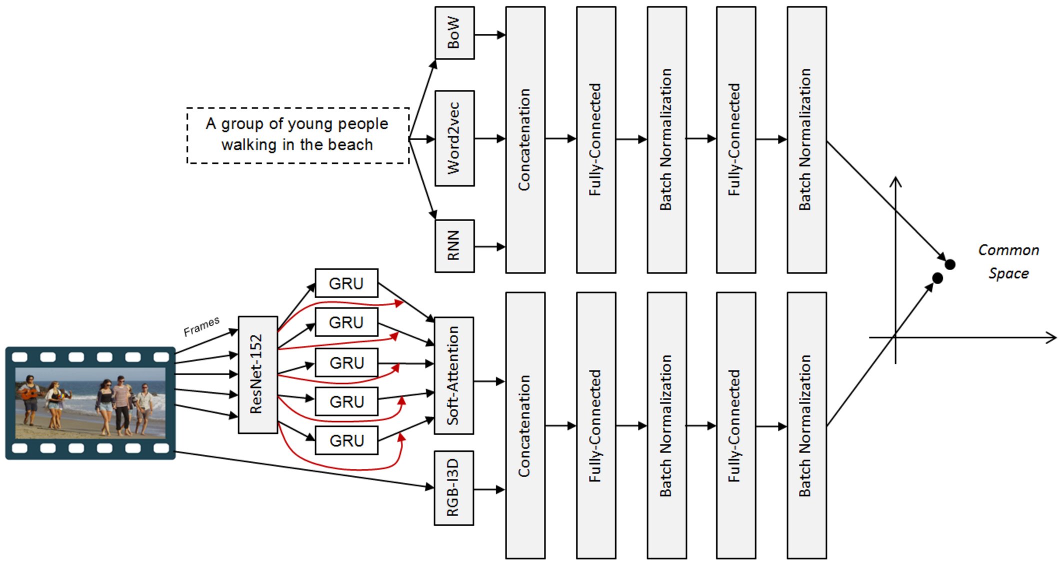


Figure 1: Our model. RNN can be a GRU, a GRC or a bidirectional GRU.

2.2 Training

We trained our model using a common hard-negative triplet ranking loss [9]. More formally, if v is the embedding computed for a video, s the embedding computed for a sentence corresponding to that video, then the loss $l(v, s)$ corresponding to the couple (v, s) is defined as follows:

$$l(v,s) = \max_{\bar{s} \neq s}(\max(0, \alpha - \cos(v,s) + \cos(v,\bar{s}))), \quad (1)$$

where α is a hyperparameter that we set to 0.2.
We used several datasets for training and validation:

- MSVD [5];
- MSR-VTT [16];
- TGIF [12] (for computer memory problems, we only used 60,000 sentence-video pairs from TGIF);
- TrecVid VTT 2016 test data [3];
- TrecVid VTT 2017 test data [2].

Table 1: Our results in terms of Mean Inverse Rank

Runs	**Subset A**	**Subset B**	**Subset C**	**Subset D**	**Subset E**
Run 1	0.194	0.190	0.194	0.193	0.199
Run 2	0.197	0.197	0.197	0.184	0.204
Run 3	0.202	0.209	0.206	0.186	0.212
Run 4	0.231	0.240	0.234	0.224	0.241

Our validation set was composed of 200 videos from TrecVid VTT 2016 test data and 200 videos from TrecVid VTT 2017 test data with corresponding sentences. Therefore, the validation set contained 400 different videos. We used MSVD, MSR-VTT and TGIF for training, for a total of 65,782 different videos with all corresponding sentences. Eventually, we used the remaining data from TrecVid VTT 2016 and TrecVid VTT 2017 to form a finetuning dataset of 3,088 videos.

We trained our models using the RMSProp method [15] with TensorFlow default parameters and gradient clipping between -5 and 5. We first trained our model on the training set, applying a learning rate of 0.00003 during 20 epochs (dividing the learning rate by two if validation loss did not improve during three consecutive epochs), with mini-batches of size 25. Then, we set the learning rate to 0.00002, and finetuned our model on the finetuning dataset during 60 epochs, dividing the learning rate by two if validation loss did not improve during three consecutive epochs.

3 Our runs

EURECOM submitted four different runs to the VTT Sentence Matching subtask. The runs are numbered from 1 to 4, with the expected best runs having the highest numbers. For each run and for each video, sentences are ranked by decreasing cosine similarity.

RUN 1 : We apply the model we described in Section 2. The RNN we used for computing sentence embeddings was a simple GRU.

RUN 2 : This run was similar to RUN 1, but we replaced the GRU by a GRC.

RUN 3 : In this run, the GRU of RUN 1 is replaced by a bidirectional GRU.

RUN 4 : This final run is a merge of previous runs. The merge is performed by summing the cosine similarities of the three previous runs, to obtain a new score for each sentence.

4 Results

We reported our results in Table 1. Our runs are ranked as we expected on subsets A, B, C and E. It is not the case for subset D, as the simple GRU obtained better results than the GRC and the bidirectional GRU.

In Figures 2-6, all results on different sentences subsets are presented. Our results are in red. As one can see, our ensemble method did better than other methods. Our future work will be dedicated to finding finer ensemble methods to see how results can be further improved.

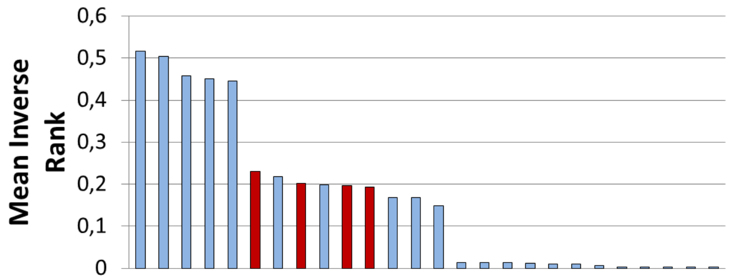


Figure 2: Results on subset A

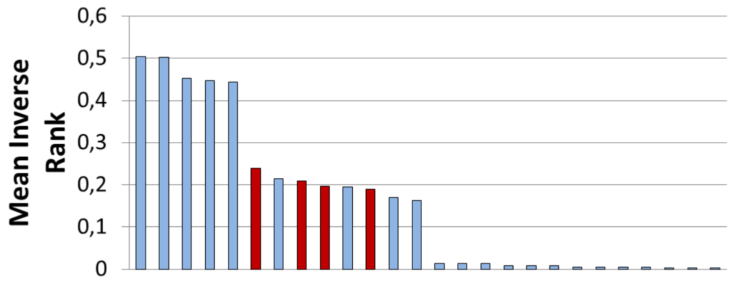


Figure 3: Results on subset B

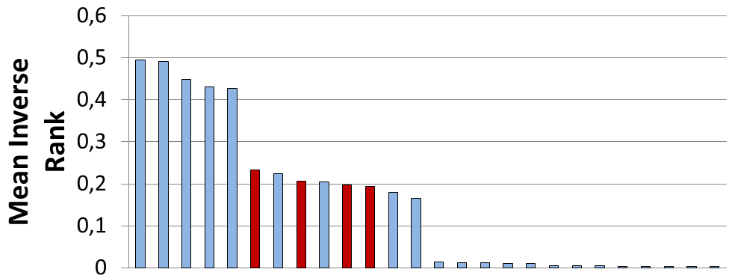


Figure 4: Results on subset C

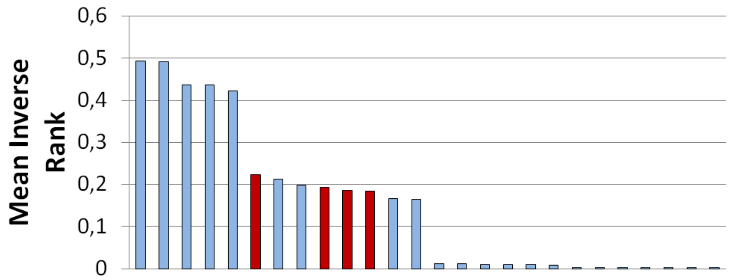


Figure 5: Results on subset D

4

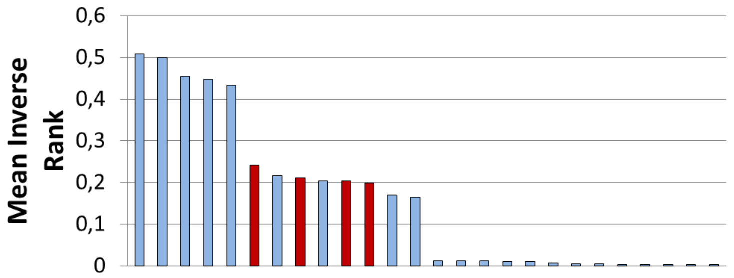


Figure 6: Results on subset E

Acknowledgments

This work has been funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No 780069 Methods for Managing Audiovisual Data: Combining Automatic Efficiency with Human Accuracy (MeMAD). We gratefully acknowledge the support of NVIDIA Corporation with the donation of one of the Titan Xp GPU used for this research

References

[1] Awad, G., Butt, A., Curtis, K., Lee, Y., Fiscus, J., Godil, A., ... & Blasi, S. (2018). TRECVID 2018: Benchmarking Video Activity Detection, Video Captioning and Matching, Video Storytelling Linking and Video Search. In Proceedings of TRECVID 2018.

[2] Awad, G., Butt, A., Fiscus, J., Joy, D., Delgado, A., Michel, M., ... & Eskevich, M. (2017, November). Trecvid 2017: Evaluating ad-hoc and instance video search, events detection, video captioning and hyperlinking. In Proceedings of TRECVID (Vol. 2017).

[3] Awad, G., Fiscus, J., Michel, M., Joy, D., Kraaij, W., Smeaton, A. F., ... & Ordelman, R. (2016). TRECVID 2016. Evaluating Video Search, Video Event Detection, Localization and Hyperlinking.

[4] Carreira, J., & Zisserman, A. (2017, July). Quo vadis, action recognition? a new model and the kinetics dataset. In Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on (pp. 4724-4733). IEEE.

[5] Chen, D. L., & Dolan, W. B. (2011, June). Collecting highly parallel data for paraphrase evaluation. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1 (pp. 190-200). Association for Computational Linguistics.

[6] Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) (pp. 1724-1734).

[7] Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009, June). Imagenet: A large-scale hierarchical image database. In Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on (pp. 248-255). Ieee.

[8] Dong, J., Huang, S., Xu, D., & Tao, D. DL-61-86 at TRECVID 2017: Video-to-Text Description.

[9] Faghri, F., Fleet, D. J., Kiros, J. R., & Fidler, S. (2017). Vse++: Improved visual-semantic embeddings. arXiv preprint arXiv:1707.05612, 2(7), 8.

[10] Francis, D., Huet, B., & Merialdo, B. (2019, January). Gated Recurrent Capsules for Visual Word Embeddings. In International Conference on Multimedia Modeling. Springer, Cham.

[11] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[12] Li, Y., Song, Y., Cao, L., Tetreault, J., Goldberg, L., Jaimes, A., & Luo, J. (2016). Tgif: A new dataset and benchmark on animated gif description. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 4641-4650).

[13] Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., ... & Zitnick, C. L. (2014, September). Microsoft coco: Common objects in context. In European conference on computer vision (pp. 740-755). Springer, Cham.

[14] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems (pp. 3111-3119).

[15] Tieleman, T., & Hinton, G. (2012). Lecture 6.5 — RmsProp: Divide the gradient by a running average of its recent magnitude. In COURSERA: Neural Networks for Machine Learning.

[16] Xu, J., Mei, T., Yao, T., & Rui, Y. (2016). Msr-vtt: A large video description dataset for bridging video and language. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 5288-5296).

A.2 AALTO's TRECVID 2018 workshop paper [2]

This paper summarises the experiments of AALTO's PicSOM group in TRECVID 2018's Video to Text (VTT) task. It shows the development of the DeepCaption image and video captioning software before and after the TRECVID 2018 submission time.

PicSOM Experiments in TRECVID 2018

Workshop notebook paper — draft

Mats Sjöberg[+], Hamed R. Tavakoli[+], Zhicun Xu[*], Héctor Laria Mantecón[+], Jorma Laaksonen[+]
[+]Department of Computer Science
Aalto University School of Science
P.O.Box 15400, FI-00076 Aalto, Finland

[*]Department of Signal Processing and Acoustics
Aalto University School of Electrical Engineering
P.O.Box 12200, FI-00076 Aalto, Finland

firstname.lastname@aalto.fi

Abstract

This year, the PicSOM group participated only in the Video to Text (VTT), Description Generation subtask. For our submitted runs we used either the MSR-VTT dataset only, or MS COCO and MSR-VTT jointly for training. We used LSTM recurrent neural networks to generate descriptions based on multi-modal features extracted from the videos. We submitted four runs:

- PICSOM_1: uses ResNet features for initialising the LSTM generator, and object and scene-type detection features as persistent input to the generator which is trained on MS COCO + MSR-VTT,
- PICSOM_2: uses ResNet and object detection features for initialisation, and is trained on MS COCO + MSR-VTT, this is the only run based on our new PyTorch codebase,
- PICSOM_3: uses ResNet and video category features for initialisation, and trajectory and audio-visual embedding features for persistent features, trained on MSR-VTT only,
- PICSOM_4: is otherwise the same as PICSOM_3 except the audio-visual embedding feature has been replaced with audio class detection outputs.

The most significant difference between our runs came from expanding the original MSR-VTT training dataset by including MS COCO, which contains images annotated with captions. Having a larger and more diverse training set seems to be bring larger improvements to the performance measures than using more advanced features. This finding has been confirmed also by our post-submission experiments that we are still continuing.

I. Introduction

In this notebook paper we describe the PicSOM group's experiments for the TRECVID 2018 evaluation [1]. We participated only in the Video to Text Description (VTT) subtask for Description Generation. Our approach is a variation on the original "Show and tell" model [2], augmented with a richer set of contextual features [3]. This year we are transitioning from an old Theano-based neural captioning system to a new PyTorch-based one, mainly because the new code base will make future development and experimentation easier from a practical standpoint. Both systems have been used to produce the runs presented in this paper, and they are described in more detail in Section II. Next, we describe the features (Section III) and datasets used for training (Section IV). Our experiments, submitted runs and results are discussed in Section V and conclusions are drawn in Section VI.

II. Neural captioning models

In our experiments we have used two different Python-based software projects for caption generation. The first and older one, *NeuraltalkTheano*, uses the Theano library whereas the second and newer one, *DeepCaption*, uses the PyTorch library.

A. NeuraltalkTheano

The Theano-based neural captioning system is described in our recent paper [3], and the source code of the implementation is freely available.[1] The neural architecture is similar to the one proposed in [2], but adds several novel properties including residual connections between the LSTM layers, and the effective usage of persistent features. Persistent features are given as an additional input to the recurrent model at each step of the caption generation, while in the typical setup features are used only for initializing the LSTM generator. A full description of the method can be found in [3].

B. DeepCaption

This year we have started to develop a new PyTorch code base, also available as open source.[2] The goal is to re-implement NeuraltalkTheano in a PyTorch architecture that is more maintainable in the long run, and thus facilitate faster development and experimentation. So far we have not yet implemented all the features of NeuraltalkTheano, in particular beam search and residual connections are still missing and have thus not been used in the DeepCaption-based result presented here.

The features are translated to the hidden size of the LSTM by using a fully connected layer. We apply dropout and batch normalization [4] at this layer.

[1]https://github.com/aalto-cbir/neuraltalkTheano
[2]https://github.com/aalto-cbir/DeepCaption

III. Features

In this section we describe the visual and auditory features used in our experiments. Table I summarizes the features and their dimensionalities. In the cases when an image modality feature extraction method has been applied to a video object, we have used the middlemost frame of the video.

TABLE I
SUMMARY OF THE FEATURES USED IN OUR EXPERIMENTS.

abbr.	feature	dim.	modality
rn	ResNet	4096	image
frA	Faster R-CNN	480	image
frB	Faster R-CNN	80	image
s	SUN397	397	image
c	category	20	image
t	trajectory	5000	video
as	audioset	527	audio
mm	multimodal	2048	multimodal

A. CNN

We are using two types of CNN features, one representation for still images, including single video frames, and one for the sequence of images, i.e. videos. Both architectures are based on the ResNet [5] model.

Image features, we are using pre-trained CNN features from ResNet 101 and 152. The 2048-dimensional features from the pool5 layer average to five crops from the original and horizontally flipped images. These features have then been concatenated together and are referred to as “rn” in Table I and later in this article.

Video features, on the video level, when there is a sequence of 16 images, we are using a 3D CNN ResNet architecture with 152 layers [6], where final fully connected layers are removed to produce a 2048-dimensional feature vector. These features are fed to the language model.

B. FasterRCNN

The types of objects and their locations in the visual scene have an effect on sentence formation and influences the adjectives used in human sentences. To extract this information, we use an object detector, specifically the Faster Region-based Convolutional Neural Network (R-CNN) [7]. This network predicts the object locations as bounding boxes and object detection scores of the 80 object categories of Microsoft Common Objects in Common Context (MS-COCO) database.[3] We use these object proposals to create object location maps. We divide the image into independent m horizontal and n vertical strips. The vertical and horizontal cells will thus overlap and the total number of cells is $m + n$.

We first define a grid on the image where each of the cells, $F_c(i)$, accumulates the integral of Gaussian distributions fit to the object proposals of the class category c as

$$F_c(i) = \sum_{b_k \in BB(c)} \iint_{b_k \cap G(i)} p(b_k) N(\text{center}(b_k), \text{diag}(b_k)) \ , \quad (1)$$

[3]http://cocodataset.org/

where $BB(c)$ is the set containing bounding box object proposals for category c, $p(b_k)$ is the confidence assigned by the detector to proposal b_k, $G(i)$ is the grid cell at position i and $N(\mu, \sigma)$ are Gaussians of given mean μ and standard deviation σ. In our experiments we used $m = n = 3$ and abbreviate these $(3 + 3) \times 80 = 480$ -dimensional features as “frA”.

We can further reduce the feature size by discarding the location information completely and just encoding the object detection scores on the image level. We obtain such an 80-dimensional feature vector using the detection score for each category, and refer to it as “frB”. For brevity of notation, concatenation of “frA” and “frB” to 560-dimensional features will be abbreviated as “frAB” in the tables below.

C. Semantic concept and category features

For still images, we use a scene-type cue as a feature to the language model. We detect the scene-type using a bank of specialized visual scene detectors trained on CNN features extracted for the SUN Scene Categorization Benchmark database.[4] Scene-type classifiers are designed using Radial Basis Function Support Vector Machine (RBF-SVM) [8]. CNN features extracted from GoogLeNet [9] pre-trained on the MIT Places dataset (from the *3rd classification branch*) is used to train a separate classifier for each variant. Each classifier determines the degree of association of a given image to the 397 scene types of the SUN database. Thus, for an input image, we form a 397-dimensional feature vector consisting of these raw scene type scores in the range of $[0, 1]$. This feature vector is referred to as “s” in Table I and below.

We also utilize the video category information, available in the MSR-VTT, as a one-hot feature vector of 20 dimensions. For the test set we have generated the corresponding feature by training a set of 20 category detectors and using the MSR-VTT category information as training data. For this purpose, we again used RBF-SVM classifiers and GoogLeNet features extracted using an ImageNet pre-trained model. This feature vector is referred to as “c”.

D. Trajectory features

For encoding genuinely video content, we use trajectory features. Dense trajectories [10] and their histogram of oriented gradients (HOG), histogram of optical flow (HOF), and motion boundary histogram x and y directions (MBHx and MBHy) descriptors are first extracted for the entire video. These five features are separately encoded into a fixed-size vector using a bag-of-features encoding with a codebook of 1,000 vectors. Each codebook was obtained using k-means clustering on 250,000 random trajectory samples from the training set. Finally, concatenating the vector encodings of each of the descriptors results in a video feature vector of 5000 dimensions. This feature vector is referred to as “t” in our tables.

[4]https://groups.csail.mit.edu/vision/SUN/

E. Audio features

The provided audio features are the occurrence probabilities for the 527 classes of the Google AudioSet Ontology[11]. AudioSet contains over 2 million 10-second human-labeled soundtracks segmented from YouTube videos. Each soundtrack can have multiple labels such as "acoustics guitar" or "door bell". Instead of providing the original audio files, AudioSet gives compact 128-dimensional embeddings which are the output of a modified VGG model, namely VGGish, for the log-mel spectrogram of audio with around a length of one second. Thus the dimension of the training data is 10×128 after being fed into the VGGish.

Inspired by the work [12] and [13], we built a similar multi-level attention model for the 10-second audio classification and achieved a mean average precision of 0.344. Since the length of the audio files in TRECVID are around 6 seconds, we decided to concatenate the same audio twice or more times and then truncated the extra parts to match the 10-second requirements. Finally, the modified 10-second audios are fed into a multi-level attention model to get the probabilities. This feature vector is referred to as "as" in Table I and below.

F. Multimodal features

To encode audiovisual information, we adopted a joint embedding space based on the ResNet [5] deep neural architecture. The architecture has two branches, one for audio and one for video. The audio branch consists of 7 residual convolution blocks and accepts a log spectrogram as input. The log spectrogram, corresponding to one second of audio data, is processed into a vector of 2048 prior to the combination with the video data by going through two fully-connected layers. On the video branch, we use the video features described in Section III-A. The output of the audio branch is used to re-weight the video branch weights. These combined responses are then fed into another fully connected layer, producing a vector of size 2048.

For training we borrow the weights for the video branch from pre-trained models of [6], which are trained on the kinetics database [14]. For the audio part, we train our network on the speech commands database [15] and then, borrow the weights from this model. The audio-visual embedding is then trained on an auxiliary task, namely action recognition. Afterwards, the obtained feature representations from this new architecture, referred to as "mm" in Table I, are used for multimodal caption generation.

IV. Training data

Here we describe the datasets used for training our captioning models. Table II gives a summary of the databases and the features we have extracted for them. In Tables II and III, we have shortened the dataset names with one letter abbreviations.

A. COCO

The *Microsoft Common Objects in COntext (MS COCO)* dataset [16] has 2,500,000 labeled instances in 328,000 images, consisting on 80 object categories. COCO is focused on non-iconic views (or non-canonical perspectives) of objects, contextual reasoning between objects, and precise 2D localization of objects.

We used COCO either with only 2014 training data "C" or with also 2014 validation data "Ca".

TABLE II
Summary of the training datasets used in our experiments.

	dataset	items		captions	features
C	COCO	82,783	img	414,113	rn frAB s
Ca	COCO-all	123,287	img	616,767	rn frAB s
M	MSR-VTT	6,513	vid	130,260	rn frAB s c t as mm
T	TGIF	125,713	vid	125,713	rn frAB s
V	MSVD	1,969	vid	80,800	rn frAB s
L	LSMDC	108,536	vid	108,536	rn frAB s

B. MSR-VTT

The *MSR-Video to Text (MSR-VTT)* dataset [17] provides 10,000 web video clips with 41.2 hours and 200,000 clip-sentence pairs in total, covering a comprehensive list of 20 categories and a wide variety of video content. Each clip was annotated with about 20 natural sentences. Additionally, the audio channel is provided too. It is intended to foster spatio-temporal information modelling and pooling strategies in video data, as well as make a broader range of domains available as opposed to previous datasets.

C. TGIF

The *Tumblr GIF (TGIF)* dataset [18] contains 100,000 animated GIFs and 120,000 natural language sentences. This dataset aims to provide motion information involved between image sequences (or frames). Authors explain that focusing on a limited series of still frames, often without narrative or need for context, and always without audio is an easier step towards full video understanding.

D. MSVD

The *Microsoft Research Video Description Corpus (MSVD)* [19] consists of 85,000 English video description sentences and more than 1,000 for a dozen more languages. Gathering efforts for this dataset presented early crowd-sourcing methodologies for video annotation. It contains a set of 2,089 videos, showing a single, unambiguous action or event. Additionally, descriptions of the same video segment can then be used as translation data if they are in different languages.

E. LSMDC

The *Large Scale Movie Description Challenge (LSMDC)* [20] 2015 consists of 108,536 short, approximately 10 second snippets of movies. Each of the snippets is associated with a genuine audio description (AD) line aimed for visually handicapped audiences and extracted from the DVD. This dataset has all references to named persons replaced with "SOMEONE" to avoid some problems in evaluating the accuracy of the generated captioning.

V. Experiments and results

During the development stage, we ran a number of experiments to select the best combinations of features and training data. We evaluated our results using the previously released TRECVID VTT 2016 test set. For the NeuraltalkTheano system (abbreviated as "nt" in Table III), the beam size in the caption generation stage was varied and it was found out that the models consistently performed best with beam size equal to one. For this reason we did not yet implement beam search for the DeepCaption system (abbreviated as "dc" in Table III), instead we used a simple greedy selection approach equivalent to having beam size one.

With the NeuraltalkTheano system we first tried using either only the COCO training data or only the MSR-VTT training data. These are seen as runs "b3" and "b4" in Table III, respectively. With the MSR-VTT training data we were also able to experiment with the multimodal "mm" and audioset "as" features, which was not possible with the image-only COCO data. These experiments are seen as submissions "s3" and "s4" in the result table, respectively. The best results with the 2016 testing data were obtained when the COCO and MSR-VTT data were used together by using only the visual modality and the middlemost video frames of the latter dataset. We regarded this as our overall best result when evaluated as the CIDEr score on the 2016 testing data and submitted it as "s1". For all our NeuraltalkTheano experiments we varied the combinations of the used features for the LSTM model initialization and for the persistent features. The feature combinations shown in Table III provided the best results on the 2016 testing data. We also varied the beam search size, but without an exception, the best results were always obtained with the beam size equal to one.

Based on evaluation on the TRECVID 2016 test set, we ended up using a 2-layer LSTM for DeepCaption with an embedding vector size of 512 and 1024 for the hidden state dimensionality. Exceptionally the runs "a1" and "a2" use 1024 for the embedding vector, which seems motivated as well by the larger dimensionality of the input features. Both in the input translation layer and in the LSTM we applied a dropout of 0.5. We used centered RMSprop [21] with a learning rate of 0.001 and weight decay (L2 penalty) of 10^{-6}.

All experiments are briefly summarized and their results presented in Table III. The four "setup" columns specify the captioning model (nt=NeuraltalkTheano, dc=DeepCaption), the initializing and persistent features, and the datasets used in the LSTM model training.

The features are concatenations of the following:

rn	= ResNet, see III-A
frA	= Faster R-CNN 480-dim, see III-B
frB	= Faster R-CNN 80-dim, see III-B
s	= SUN397, see III-C
c	= Category of 20 video genres, see III-C
t	= Trajectory, see III-D
as	= Audioset, see III-E
mm	= MultiModal, see III-F

The used datasets are concatenations of the following datasets, each described in one of the subsections of the previous section:

C	= COCO, see IV-A
M	= MSR-VTT, see IV-B
T	= TGIF, see IV-C
V	= MSVD, see IV-D

Our results compared to those of the other submitted runs are visualized with bar charts for each automatic performance measure in Figures 1–5.

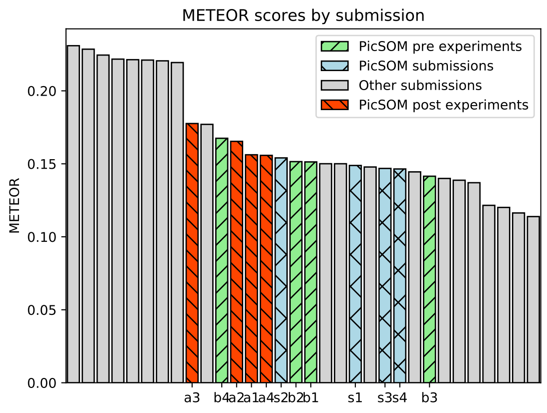


Fig. 1. METEOR results of our group and others.

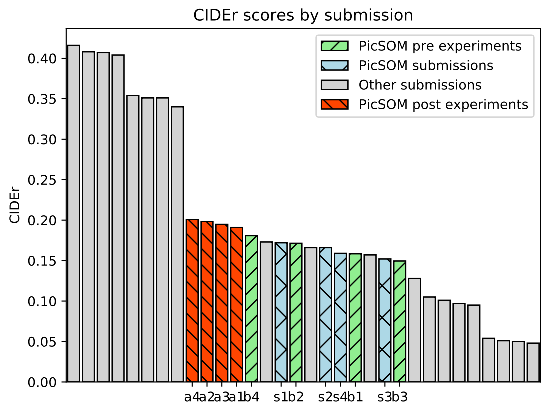


Fig. 2. CIDEr results of our group and others.

VI. Conclusions

The most practical conclusion for our group's internal use is that we have been successful in replacing our old Theano-based captioning system with a new PyTorch-based one that has overtaken the old system in performance.

Compared to the level of performance reached by some of the other research groups we are, however, still clearly behind. We will still need to continue our efforts to improve the evaluation scores obtained by our DeepCaption system.

TABLE III

Results of our submissions (s1,..., s4) and some noteworthy pre (b1,..., b4) and post (a1 and a2) experiments.

	setup				2016		2018				
id	**mod**	**init**	**pers**	**data**	**METEOR**	**CIDEr**	**METEOR**	**CIDEr**	**CIDErD**	**BLEU**	**STS**
b1	dc	rn	–	C+M	0.2135	0.2620	0.1513	0.1584	0.0471	**0.0110**	
b2	dc	rn+frAB	–	C+M	**0.2186**	**0.2872**	0.1515	0.1714	0.0475	0.0082	
b3	nt	rn+c	t	M	0.2005	0.2379	0.1415	0.1495	0.0364	0.0051	
b4	nt	rn	frAB+s	C	0.1890	0.1907	**0.1675**	**0.1808**	**0.0641**	0.0091	
s1	nt	rn	frAB+s	C+M	0.2147	**0.2886**	0.1488	**0.1720**	0.0450	0.0053	**0.3806**
s2	dc	rn+frB	–	C+M	**0.2214**	0.2750	**0.1540**	0.1660	**0.0480**	**0.0091**	0.3739
s3	nt	rn+c	t+mm	M	0.2039	0.2437	0.1468	0.1520	0.0390	0.0055	0.3676
s4	nt	rn+c	t+as	M	0.2021	0.2413	0.1464	0.1590	0.0400	0.0048	0.3713
a1	dc	rn+frAB+s	–	C+M+T	0.2238	**0.3158**	0.1562	0.1910	0.0525	0.0122	
a2	dc	rn+frAB+s	–	C+M+T+V	0.2300	0.3080	0.1654	0.1984	0.0653	0.0166	
a3	dc	rn	–	C+T	**0.2343**	0.2997	**0.1776**	0.1948	**0.0700**	**0.0197**	
a4	dc	rn	–	Ca+M+T+V	0.2191	0.3070	0.1558	**0.2007**	0.0543	0.0169	

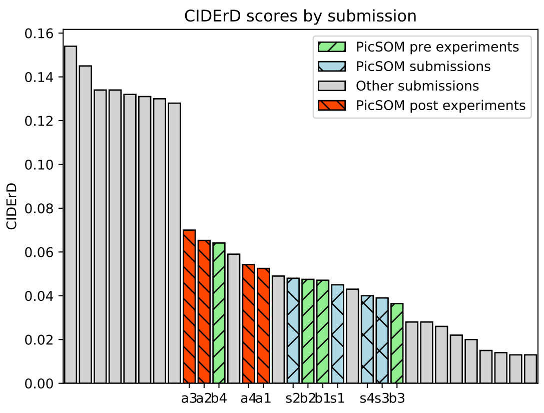


Fig. 3. CIDErD results of our group and others.

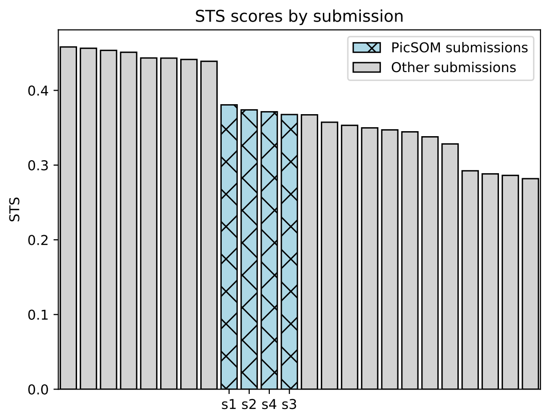


Fig. 5. STS results of our group and others.

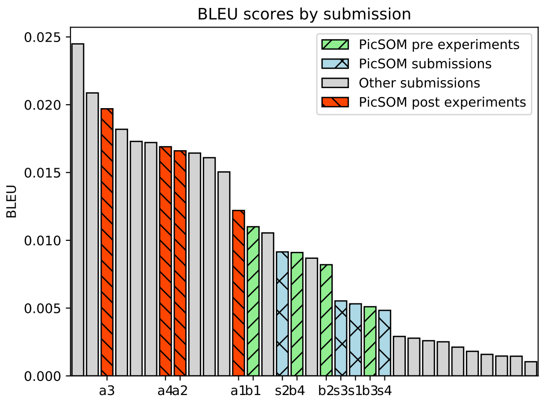


Fig. 4. BLEU results of our group and others.

Comparing the results we obtained by using the 2016 VTT test data and those obtained with this year's test data, we can draw two conclusions. First, this year's results seem to be clearly worse than those of 2016. This will need further studies as the number of reference captions has increased from two to five, which in general should have had the opposite effect on the results. Second, it seems that we were not able to choose the best-performing models among the variants we experimented with, based on the evaluation scores with the 2016 data.

The pre-submission results obtained by using only the COCO training data now seem to be very competitive. In general, however, our results indicate that using more training data, even with only one image frame for each training set video, is more beneficial than using genuinely video, audio or multimodal features. We will continue this path of experiments further for the final version of this report.

Acknowledgments

This work has been funded by the grant 313988 *Deep neural networks in scene graph generation for perception of visual multimedia semantics* (DeepGraph) of the Academy of Finland and the European Union's Horizon 2020 research and innovation programme under grant agreement No 780069 *Methods for Managing Audiovisual Data: Combining Automatic*

Efficiency with Human Accuracy (MeMAD). The calculations were performed using computer resources provided by the Aalto University's *Aalto Science IT* project and CSC – IT Center for Science Ltd. We also acknowledge the support of NVIDIA Corporation with the donation of a Quadro P6000 GPU used for parts of this research.

References

[1] George Awad, Asad Butt, Keith Curtis, Jonathan Fiscus, Afzal Godil, Alan F. Smeaton, Yvette Graham, Wessel Kraaij, Georges Quénot, Joao Magalhaes, David Semedo, and Saverio Blasi. Trecvid 2018: Benchmarking video activity detection, video captioning and matching, video storytelling linking and video search. In *Proceedings of TRECVID 2018*. NIST, USA, 2018.

[2] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[3] Rakshith Shetty, Hamed R.-Tavakoli, and Jorma Laaksonen. Image and video captioning with augmented neural architectures. *IEEE MultiMedia*, 25(2):34–46, April 2018.

[4] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. Technical report, Microsoft Research, 2015.

[6] Kensho Hara, Hirokatsu Kataoka Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3D CNNs retrace the history of 2D CNNs and ImageNet? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[7] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, 2015.

[8] Markus Koskela and Jorma Laaksonen. Convolutional network features for scene recognition. In *Proceedings of the 22nd ACM International Conference on Multimedia*, Orlando, FL, USA, November 2014.

[9] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *arXiv*, abs/1409.4842, 2014.

[10] Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. Action recognition by dense trajectories. In *Proceedings of the 24th IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3169–3176. IEEE Computer Society, 2011.

[11] Jort F Gemmeke, Daniel PW Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R Channing Moore, Manoj Plakal, and Marvin Ritter. Audio set: An ontology and human-labeled dataset for audio events. In *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*, pages 776–780. IEEE, 2017.

[12] Qiuqiang Kong, Yong Xu, Wenwu Wang, and Mark D Plumbley. Audio set classification with attention model: A probabilistic perspective. *arXiv preprint arXiv:1711.00927*, 2017.

[13] Changsong Yu, Karim Said Barsim, Qiuqiang Kong, and Bin Yang. Multi-level attention model for weakly supervised audio classification. *arXiv preprint arXiv:1803.02353*, 2018.

[14] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset. *arXiv:1705.06950*, 2017.

[15] Pete Warden. Speech commands: A dataset for limited-vocabulary speech recognition. *arXiv:1804.03209*, 2018.

[16] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common objects in context. In *European Conference on Computer Vision (ECCV)*, 2014.

[17] Jun Xu, Tao Mei, Ting Yao, and Yong Rui. MSR-VTT: A large video description dataset for bridging video and language. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5288–5296, 2016.

[18] Yuncheng Li, Yale Song, Liangliang Cao, Joel Tetreault, Larry Goldberg, Alejandro Jaimes, and Jiebo Luo. TGIF: A new dataset and benchmark on animated gif description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4641–4650, 2016.

[19] David L. Chen and William B. Dolan. Collecting highly parallel data for paraphrase evaluation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 190–200, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[20] Anna Rohrbach, Marcus Rohrbach, Niket Tandon, and Bernt Schiele. A dataset for movie description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[21] Alex Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

A.3 INA's MIREX 2018 workshop paper [3]

This paper summarises INA's system for the MIREX 2018 music and speech detection challenge. INA's submission received the top result in the evaluation.

INA'S MIREX 2018 MUSIC AND SPEECH DETECTION SYSTEM

David Doukhan **Eliott Lechapt** **Marc Evrard** **Jean Carrive**
French National Institute of Audiovisual (Ina), Paris, France
{ddoukhan,elechapt,mevrard,jcarrive}@ina.fr

ABSTRACT

A convolutional neural network (CNN) based architecture is proposed for MIREX 2018 music and speech detection challenge. The system uses log-mel filterbank features. It has 4 convolutional and 4 dense layers. It is part of the inaSpeechSegmenter open-source framework, which was designed for conducting gender equality studies.

1. INTRODUCTION

This paper presents the system submitted to Mirex 2018 Speech and/or Music detection task. This system is based on the `inaSpeechSegmenter` open-source framework (MIT license) [4]. The full framework is available on GitHub [1] and is packaged as a python3 pip module [2].

It was designed for conducting digital humanities studies describing men and women speaking-time ratios across TV and radio channels. These large-scale descriptions were used as an estimate of gender equality in medias [3, 5].

The framework is composed of two segmentation modules. The first module, which was used for the speech and/or music detection task, is in charge of segmenting audio stream into speech and music. The system was designed for segmentation rather than detection. With respect to the aim of this framework (estimating men and women speech-time ratio), speech-over-music is labeled as speech. Therefore, this module is better suited for speech than for music detection. The second module of the framework (not evaluated in this challenge) is in charge of splitting and labeling the resulting speech segments according to their corresponding gender class.

2. ALGORITHM

The processing pipeline is composed of 4 main steps described in the following subsections.

2.1 Signal Activity detection

A baseline activity detection system – based on adaptive energetic threshold – is used. The threshold is defined as the $\log(3\%)$ of the mean log energy found in a given sound file. Filtering procedures are then used to obtain the segments showing activity from these frame-level activity estimates.

[1] `https://github.com/ina-foss/inaSpeechSegmenter`
[2] `https://pypi.org/project/inaSpeechSegmenter/`

2.2 Feature extraction

The Mel-scaled filter-banks representation of the signal is computed on 25ms sliding windows with 10 ms shift using SIDEKIT [7]. 21 Mel filterbanks sampled between 100 and 4000 Hz are extracted. This rather low maximum frequency limit was chosen to handle low-quality signals that may occur in TV and radio streams: i.e., telephone-quality signal with no energy above 4000 Hz.

Resulting features were grouped into patches concatenating 68 adjacent windows. Patches were normalized to have 0 mean and unit variance, in order to increase the robustness to volume variations between recordings.

2.3 CNN frame-level detection

Figure 1 shows the CNN architecture used for speech-music discrimination, which was implemented using Keras [2]. The CNN input is composed of patches of dimension 68×21 (time $\times$ feature dimension), accounting for an input analysis window of 695 ms. The model has 4 convolutional and 4 dense layers. All these layers are followed by a batch-normalization stage and ReLU activation layers. The dense layers are also followed by dropout layers, with dropout rates increasing according to the network depth. The first convolutional layer is associated with the biggest width (5), in order to capture the horizontal patterns typically found in music signals. The last pooling layer operates on the temporal dimension, in order to focus on the pattern showing the biggest activation in this relatively large time-interval. The output is implemented using a softmax activation layer, allowing to obtain a probability estimated for each supported class (i.e., speech, music).

2.4 Viterbi decoding

The CNN output is composed of instantaneous frame-level probabilities for speech and music. These probabilities are used to feed a 2-states Hidden Markov Model, aimed at inferring the most likely sequence of hidden states from these frame-level estimates. State transition probabilities were defined empirically through a grid-search procedure performed on development datasets (section 4).

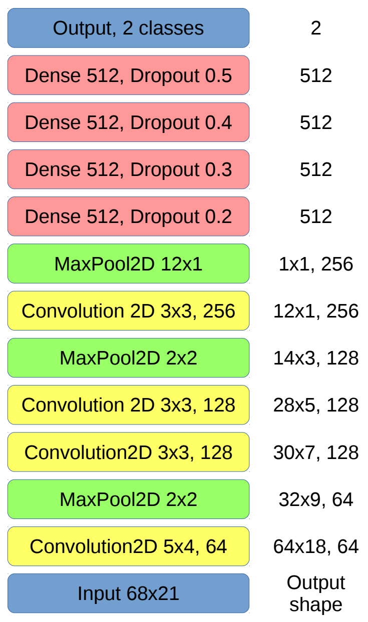


Figure 1. CNN speech-music classification architecture.

3. TRAINING DATASETS

Several datasets aimed for classification were used to train the proposed convolutional neural network model. Three datasets dedicated to speech versus music classification were used: GTZAN [11], Scheirer-Slaney [9] and MUSAN [10]. Additional music data was obtained from the GTZAN musical genre corpus [11]. 72 excerpts corresponding to *a cappella* singing, which was identified as a difficult musical genre, were obtained using the free music archive API[3]. Lastly, we used Ina's speaker dictionary to increase speech data with 2300 distinct speakers [8, 12].

Data labelled as speech in GTZAN, Scheirer-Slaney and Ina's speaker dictionary was possibly mixed with music, while data labelled as music does not contain speech. Therefore, the corresponding classification system is able to discriminate speech, including speech over music, from music alone.

4. DEVELOPMENT DATASETS

The models most suited to frame-level classification were not necessarily suited to the segmentation and the detection tasks. These were evaluated and tuned to optimize frame-level f-measure on 2 distinct detection datasets.

We used the REPERE challenge dataset, which was designed for speech transcription and speaker identification tasks [6]. This dataset does not distinguish between speech and speech-over-music.

Lastly, we used the Muspeak sample data provided for the MIREX-2015 speech/music detection challenge [1].

[3] `http://freemusicarchive.org`

5. ACKNOWLEDGEMENTS

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 780069 (MeMad project).

6. REFERENCES

[1] Mirex 2015 music/speech classification and detection and challenge. Visited on 2017-03-07.

[2] François Chollet et al. Keras. `https://github.com/fchollet/keras`, 2015.

[3] David Doukhan and Jean Carrive. Description automatique du taux d'expression des femmes dans les flux télévisuels français. In *XXXIIe Journées d'Études sur la Parole*, pages 496–504, 2018.

[4] David Doukhan, Jean Carrive, Félicien Vallet, Anthony Larcher, and Sylvain Meignier. An open-source speaker gender detection framework for monitoring gender equality. *Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

[5] David Doukhan, Géraldine Poels, and Jean Carrive. Describing gender equality in french audiovisual streams with a deep learning approach (accepted). *Journal of European Television History and Culture (VIEW)*, 2018.

[6] Aude Giraudel, Matthieu Carré, Valérie Mapelli, Juliette Kahn, Olivier Galibert, and Ludovic Quintard. The repere corpus: a multimodal corpus for person recognition. In *LREC*, pages 1102–1107, 2012.

[7] Anthony Larcher, Kong Aik Lee, and Sylvain Meignier. An extensible speaker identification sidekit in python. In *Acoustics, Speech and Signal Processing (ICASSP)*, pages 5095–5099. IEEE, 2016.

[8] François Salmon and Félicien Vallet. An effortless way to create large-scale datasets for famous speakers. In *LREC*, pages 348–352, 2014.

[9] Eric Scheirer and Malcolm Slaney. Construction and evaluation of a robust multifeature speech/music discriminator. In *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97.*, volume 2, pages 1331–1334, 1997.

[10] David Snyder, Guoguo Chen, and Daniel Povey. Musan: A music, speech, and noise corpus. *arXiv preprint arXiv:1510.08484*, 2015.

[11] George Tzanetakis and Perry Cook. Musical genre classification of audio signals. *IEEE Transactions on speech and audio processing*, 10(5):293–302, 2002.

[12] Félicien Vallet, Jim Uro, Jérémy Andriamakaoly, Hakim Nabi, Mathieu Derval, and Jean Carrive. Speech trax: A bottom to the top approach for speaker tracking and indexing in an archiving context. In *LREC*, 2016.

A.4 AALTO's submitted journal article [4]

This paper describes AALTO's state-of-the-art hybrid HMM-DNN ASR methods that are particularly useful for developing ASR systems for low-resource languages with complex morphology. The system is evaluated here for Finnish, Arabic, Swedish and English.

Noname manuscript No.
(will be inserted by the editor)

Advances in Subword-based HMM-DNN Speech Recognition Across Languages

Peter Smit[1,2] · Sami Virpioja[1,3] · Mikko Kurimo[1]

Received: date / Accepted: date

Abstract We describe a novel way to implement subword language models in speech recognition systems based on weighted finite state transducers, hidden Markov models, and deep neural networks. The acoustic models are built on graphemes in a way that no pronunciation dictionaries are needed, and they can be used together with any type of subword language model, including character models. The advantages of short subword units are good coverage, reduced data sparsity, and avoiding vocabulary mismatches in adaptation. Moreover, constructing neural network language models (NNLMs) is more practical for small input and output layers. We also propose methods for combining the benefits of different types of language model units by reconstructing and combining the recognition lattices. We present an extensive evaluation of various subword units on speech datasets of four languages: Finnish, Swedish, Arabic, and English. The results show that the benefits of short subwords are even more consistent with NNLMs than with traditional n-gram language models. Combination across different acoustic models and language models with various units improve the results further. For all the four datasets we obtain the best results published so far. Our approach performs well even for English, where the phoneme-based acoustic models and word-based language models typically dominate: Combination of several grapheme-based models provides 4% improvement over phoneme-based baseline, and combining both grapheme and phoneme models yields the state-of-the-art error rate of 15.9%

1
Department of Signal Processing and Acoustics
Aalto University, Espoo, Finland
E-mail: firstname.lastname@aalto.fi

2
Inscripta, Helsinki, Finland

3
Utopia Analytics, Helsinki, Finland

for the MGB 2018 dev17b test. For all languages we also show that the language models perform reasonably well with limited training data.

Keywords Large vocabulary speech recognition · Subword units · Character units · Recurrent neural network language models

1 Introduction

The term large vocabulary continuous speech recognition typically refers to speech recognizers operating on tasks that have a broad domain and cover the majority of regularly used words. Implicitly, it also assumes that a speech recognition system has a vocabulary which must be large to sufficiently cover the language used. Originally, the term also implies that the vocabulary consists of words, similarly as how the vocabulary is treated in linguistics, except that often the surface forms of the words are used explicitly.

Unfortunately, even the largest vocabularies limit the use of words into the listed ones. However, it is not possible to create an exhaustive list of words, because all training data is also limited and languages evolve with new words appearing and old words being forgotten. In speech recognition, the coverage of the vocabulary is often measured by the out-of-vocabulary rate. It is the proportion of words that are not present in the vocabulary and thus cannot be recognized correctly.

For morphologically rich languages, such as Finnish, attempts to make a vocabulary to cover a sufficient part of the language is even more problematic. Word formation by derivation and compounding produces a massive number of lexical words, and inflectional processes further create a large number of variations requiring dozens or more surface forms to cover a single lexical word.

In the context of these problems, subword-based speech recognizers have been developed to cope with unlimited (or open) vocabulary tasks [5, 14]. Instead of words, the vocabulary contains morphemes, syllables, or other subword units that together can be used to create an unlimited amount of word forms. If the units have been appropriately chosen, all words in the language can be generated and modeled by the system. This would include words not seen in the training data or even words that might not even have existed yet at the time the system is created. One step further is to create a vocabulary-free system based only on characters. These systems have full freedom to predict any word, as long as it can be written by the characters known to the system.

A subword or vocabulary-free system also has drawbacks. Some languages, such as English, have strong pronunciation variations in words and character sequences, which are hard to model on the subword or character level. While this could be solved by finding and introducing the relevant pronunciation variants in the pronunciation dictionary for subwords, we take here a phoneme-free approach and build the acoustic models directly for graphemes. Another drawback is that during decoding, the unlimited vocabulary systems also consider many non-words and thus have a larger search space.

Besides in Hybrid HMM-DNN based systems, the character and subword-based language models are also relevant for the Connectionist Temporal Classification (CTC) or other ‘end-to-end‘ style models. Although the latest end-to-end systems perform very well, the Hybrid HMM-DNN systems are often still the preferred choice. This is, for example, the case when separate language models are needed, or when much more text data is available than audio data. Especially in low-resource situations this is often the case. However, the end-to-end systems do use subword and charater-based models for much of the same reasons, including the size of layers in the neural networks as well as the more effective use of training data. Like in our systems, the grapheme-based acoustic models are also relevant for end-to-end models and often are the preferred choice [33,31].

In the following sections, we describe how we created effective subword and character based models in a conventional speech recognition system, where the acoustic and language models are trained separately. We introduce the concepts needed to implement subwords correctly in a weighted finite state transducer (WFST) based recognizer and the considerations needed to train both conventional n-gram language models as well as modern recurrent neural network language models. Besides merely testing systems with different units, we also propose, implement, and evaluate system combinations that combine models composed of different units.

The experiments are repeated on benchmarks covering four different languages from three different language families with different vocabulary sizes and morphological complexities. For all of these four speech datasets we obtain, to the best of our knowledge, the best results published so far. The motivation for this work came from the improvements we obtained by developing the new subword WFSTs [37], character-based language models [36] and then winning the 2017 multi-genre broadcast speech recognition challenge by combining systems operating on different subwords [35]. While some of the ideas were initially presented in these conference papers, they have now been extended and analyzed in detail here. All the results are entirely new and the tasks include also Swedish and English, for which such methods have not been proposed before.

In summary, the main novelty in this work is the set of tools and techniques for successful subword modeling for WFST-based hybrid DNN-HMM speech recognition using graphemes instead of phonemes. In addition, it includes a thorough evaluation across four diverse languages as well as an evaluation of these techniques in an under-resourced scenario. Moreover, it explores the usage and considerations in using subword and character based neural-network language models for hybrid DNN-HMM systems. Lastly, it introduces and evaluates the tools for doing lattice combination across different language modeling units, allowing, for example, Minimum Bayes Risk decoding with a larger variety of models.

Table 1 Four methods of marking subword units so that the original word sequence 'two slippers' can be reconstructed

Style (abbreviation)	Example
boundary tag (<w>)	<w> two <w> slipp er s <w>
left-marked (+m)	two slipp +er +s
right-marked (m+)	two slipp+ er+ s
left+right-marked (+m+)	two slipp+ +er+ +s

2 Subword modeling for ASR

Subword modeling has been used for almost twenty years in speech recognition. For some languages, such as Arabic, it has been popular to use linguistic units such as morphemes as the basic language modeling unit [7,18,26]. For other languages, such as Finnish, the subword segments are often created with data-driven methods [14,8]. Multiple data-driven methods for subword segmentation have been used in speech recognition. In [37] we have shown that if the parameters and context length for the language model are optimized to a comparable level, the actual segmentation method has only a minor effect on the speech recognition performance.

Not only do these systems need a subword segmentation, but also to reconstruct words back from the subword units. This can be done by e.g. adding a dummy unit to mark a word boundary, or by creating different subword variants based on the location in a word. The first subword systems often used a separate word boundary marker [15] or a continuation marker attached on the left side of the subword when there is no word boundary [4,40]. In [37] we showed that the selected marking style can actually have a profound effect on the speech recognition result and that the optimal marking style often depends on the data set. Table 1 shows a list of the common marking styles, as well as the corresponding abbreviations used in this paper.

After segmenting the training texts into subwords, conventional tools can be used for training the language models by treating the marked subwords as independent words. Only when the word-based perplexity is calculated, the actual reconstructed words need to be accounted for. If the word boundary tags (<w>) are used for that, these tags will be treated as normal tokens which have their own probability. This is necessary to predict the word boundaries correctly, otherwise any location of word boundary tags would be as likely.

In this work we design our subword modeling in such way that it is independent from the acoustic model. The acoustic model can be trained on sequences of phones (or graphemes), and any type of language model can be used with it, whether it is using characters, other subwords, or words.

2.1 Pronunciation lexicon

In a traditional speech recognition system, the vocabulary is directly linked with the pronunciation lexicon. For languages such as English, where the pronunciation is not based on rules, often hand-crafted lexicons that map each word to the correct phoneme sequence are used. Typically, most words include only a single or only a few pronunciations.

When words are split into subwords, it is not always clear what the pronunciation of each subword is. A word can have more or fewer phonemes than graphemes, and the relation that might be obvious to humans might be hard to determine algorithmically. Many subwords will also have multiple pronunciation variants. Moreover, many subwords will share the same pronunciation sequence, which increases the complexity of the recognition process.

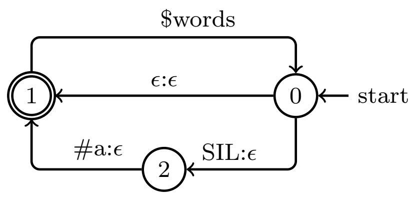
One solution to this problem is to use grapheme-based units instead of phonemes, and let the acoustic model learn the pronunciation patterns from the training data. In languages that are highly phonemic, e.g. Finnish, this is a natural thing to do. For languages that have a more irregular pronunciation pattern, e.g. English, this is more complicated, because the acoustic model must learn the pronunciation of characters based on their context. In Gaussian mixture model (GMM) based acoustic models, the phoneme-based recognizers outperform the grapheme-based ones for English by margins as big as a 50% relative increase in word error rate (WER) [23]. However, a recent work [42] shows that modern deep neural network (DNN) acoustic models can learn the English pronunciations better, and the difference between phoneme and grapheme-based recognizers can be as low as 5% relative WER. Also, the modern sequence-to-sequence trained recognizers often predict graphemes directly without using a separate pronunciation lexicon [30].

In this work, we used grapheme-based acoustic models for all languages, which solves the problem of generating pronunciations for all subwords.

2.2 Subword modeling in WFST based speech recognition

In the weighted finite state transducer (WFST) framework [24] a decoding graph is created by the composition of four different FSTs, abbreviated with the term HCLG. The H-FST maps HMM states to context-dependent phones, the C-FST maps these to context independent phones. The L-FST (lexicon) maps the phone sequences to words and the G-FST scores the sequences of words with a language model.

If there would be no other phones than those present in words, these FSTs could be used for subword systems without modifications. However, there are two common extensions made to the lexicon FST, which do have an impact on subword models. First, the lexicon FST often allows optional silence phones to be inserted between words and on the beginning and end of the utterance. However, in subword FSTs, there cannot be any silence phones between subwords that belong to the same word. Second, in words, different phoneme-variations are used depending on the location, e.g. whether the phoneme is the first or last one in a word. To take this into account, also the subword FSTs must be aware of their position in the current word.



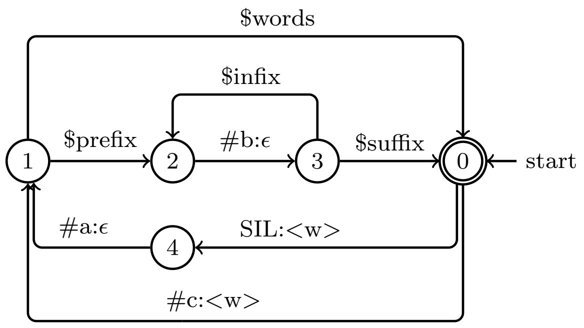


Fig. 1 The original lexicon FST (top) and the subword-modified lexicon FST (bottom). The labels on the arcs show the input:output label for each transition. ϵ represents the empty symbol. Labels starting with '#' are dummy labels that are used for keeping the FSTs deterministic (a requirement in the Kaldi toolkit). Labels that start with $ are replaced with linear FSTs containing the actual pronunciations of the applicable words.

Fig. 1 shows how the lexicon FST can be created in such a way that these properties still hold for subword-based systems. The subword lexicon is split in four categories (prefix, infix, suffix and complete words) in which the position-dependent phones are marked appropriately. Note that depending on the marking style a subword might appear in multiple categories. E.g., with the <w>-style, all subwords belong to all four categories. More details and evaluations can be found in [37].

2.3 Language modeling with subword units

The choice of language modeling unit has significant implications for a speech recognizer. Although the tools and techniques do not necessarily change—simple n-gram models will work to a certain degree—the optimization is very dependent on the chosen unit. The choice of units changes the frequential characteristics of the tokenized training text. When words are split more, the number of tokens increases and number of types decreases. The extreme case is to split into characters, where the lexicon would constitute of only the character set. When the number of tokens in a sentence increases, it increases the number of units needed to represent the required context information. In n-gram models the only way to capture the more context information with shorter units is to increase the n, the length of the preceding context. With standard toolkits this would also increase the size of the model exponentially. For this reason, we use the VariKN toolkit [34], which uses dynamic modified Kneser-Ney growing and pruning of n-gram models. The resulting models, so-called varigrams, do not fix the length of the preceding contexts but let it depend on the improvement it gives to the model. In practice, a model with characters as units might have the majority of n-grams with a context between 8-14 preceding tokens.

For recurrent neural network (RNN) based language models, it is of less importance how many tokens there are in a sentence, as the models can learn dependencies that occur between tokens further apart in history [22]. Besides that, the subword models have an advantage over word models because of their smaller vocabulary size. If the vocabulary of a system is too large, a lot of computing and training time is used for learning the parameters of the input and output layer which contain the same number of units as the vocabulary size. Multiple methods to combat this, such as class-based training [6] or hierarchical softmax [25, 20] have been proposed, but for the subword models, these methods are typically not needed as the size of the input and output layer is reasonable by default.

For both n-gram and RNN language models the most prominent advantage can be found in the increased coverage of vocabulary and the reduced sparsity of the training data. The subword models based on the units learned by the Morfessor algorithm have the ability to predict almost any word in the language, even words that were not seen in the training data and those that only came into existence after the models were trained. Also, the units of subword models occur more frequently and in more different contexts, giving the models more examples to train from. In a word-based system, more than half of the words might only appear once or twice, which is not enough to build a robust estimate of their occurrence in future texts.

3 System evaluations

To evaluate the properties of subword and character-based models in a modern neural network-based speech recognition system, we have built systems for benchmarking tasks in four different languages from three different language families. First, we evaluate for each language the general performance of a word-based model and then the performance on different subword marking styles for n-gram-based language models. After that, we select the best marking and evaluate the different levels of segmentation by changing the corresponding parameter in Morfessor. We evaluate these models both on n-gram language models and two different recurrent neural network architectures; one ‘shallow’ and one ‘deep’ neural network. The details of these systems are described in Section 3.1.

3.1 Setup

We have used the same set of tools and recipes to create the acoustic and language models for all languages. The acoustic models are trained with the

Kaldi toolkit [28]. First GMM-based models are trained, which are used to automatically clean and segment the data with the standard facilities present in Kaldi. The resulting cleaned data set is used to train a neural network based acoustic model. For all languages, we trained three different models. The most basic one is a time-delay neural network (TDNN) [27], which is a non-recurrent neural network that takes a fixed time window as input. Furthermore, the parameters of lower layers are shared between smaller time windows in a similar fashion to convolutional neural networks. The second NN-based model was a mixture of TDNN layers and long short-term memory layers (LSTM), which is a recurrent architecture that retains information from previous samples to improve its modeling power. The last model is a bi-directional TDNN-LSTM model, which also unrolls a network forward in time. In our experience, the recurrent models require more data in order to be trained adequately. Therefore, they are not expected to outperform the regular TDNN model for smaller data sets. All neural network acoustic models are trained with lattice-free MMI [29], which does not primarily optimize frame-based phone prediction accuracy, but instead decodes part of the utterance on the fly and calculates the error with the Maximum Mutual Information criterion. However, cross-entropy is still used for regularization. For all languages, we first used a simple word n-gram model to determine the best of the three optional acoustic models and then used that model through all experiments.

The different subword segmentations, besides the character one, are trained with the Morfessor 2.0 [41], using the basic unsupervised Morfessor Baseline algorithm [9]. In order to obtain models that have different vocabulary sizes, multiple values for the corpus weight parameter α are used in Morfessor. After segmenting the language model training text into subword tokens, four different unit boundary marker variants are created as explained in Section 2.2. For the language modeling toolkits, all variants can be trained using the same procedure; the only difference is that the marked subwords are given as input instead of words.

The n-gram language models are trained with the VariKN toolkit [34]. This toolkit uses modified Kneser-Ney smoothing and grows and prunes n-gram models dynamically, possibly including very long contexts if that results in a better model. In practice, the models are trained without limiting the order n. Instead, the total model size is controlled by the growing and pruning parameters. For the first recognition pass, we train a model with appr. 4-10 million n-gram contexts. For the rescoring pass, we train larger models, with no limits for growing or pruning. This means that all contexts that add anything to the prediction power will be added to the model. For word models, there remain typically between 50-80 million n-gram contexts, for character models typically up to 400 million n-gram contexts.

We train the RNN language models with the TheanoLM toolkit [11]. Both a shallow and a deep version of these networks is used; their basic architecture is shown in Fig. 2. We train the models at most 15 epochs with AdaGrad, and stop early if perplexity on the development set does not decrease. For models with a very large vocabulary, we use a class-based output using classes trained

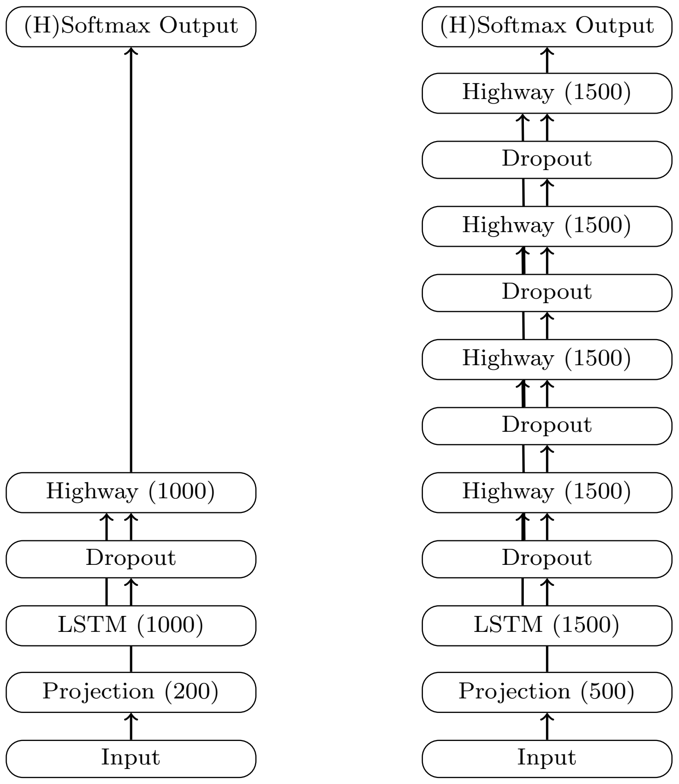


Fig. 2 The two used RNNLM architectures. Both networks start with a projection and LSTM layer [16]. On the left the 'shallow' architecture that only contains a single highway and dropout layer. On the right the 'deep' architecture that has four pairs of dropout [38] and highway [39] layers. The number between parenthesis is the number of units in the layer.

with the exchange algorithm [6, 19]. The RNN-based models are then used to rescore the n-gram-based lattices. Standard pruning algorithms such as limiting the amount of active search paths and restricting the maximum history (from 25 tokens for word models to 100 tokens for character models) are applied in a similar manner as in [12]. The 'shallow' and 'deep' architectures were only mildly optimized. The 'shallow' network was chosen as it corresponds to our previous work [36] and the 'deep' architecture was the same as in [12], inspired by [39]. We further optimized the number of parameters in the deep model.

For all experiments, we report the word error rate (WER), which uses the counts of substitutions, deletions, and insertions to the reference text to calculate the error rate.

3.2 Finnish

Finnish is an agglutinative language from the Uralic language family with a very large word vocabulary, which makes it especially suited to subword-based recognition. The written language is highly phonemic; a phoneme-based lexicon would be almost equivalent to a grapheme-based lexicon.

Table 2 Evaluations for the Finnish YLE broadcast news data set

TDNN	TDNN-LSTM	TDNN-BLSTM
19.4	18.5	**18.4**

(a) Comparison of different acoustic models using a word n-gram language model.

Segmentation	Vocab size	<w>	+m+	m+	+m
char	29	17.3	16.9	17.1	17.0
morf 0.001	9502	15.9	15.7	16.2	16.4
morf 0.01	53183	16.0	15.8	16.4	16.4
morf 0.1	248257	16.1	15.7	16.4	16.4
word	4308628	16.7			

(b) Rescored n-gram results for different subword markers and segmentations using a TDNN-BLSTM acoustic model.

Segmentation	<w>	+m+	m+	+m
char	4589 (13)	4090 (35)	4158 (13)	4270 (13)
morf 0.001	3499 (13)	3507 (35)	3486 (13)	3540 (13)
morf 0.01	3489 (21)	3522 (46)	3364 (21)	3395 (23)
morf 0.1	3607 (37)	3494 (90)	3347 (48)	3390 (49)
word	2328 (810)			

(c) n-gram perplexity results for different subword markers and segmentations on the ASR development set. The number of OOV tokens are presented between parenthesis.

		RNNLM	
Segmentation	n-gram	shallow	deep
char	16.9	14.4	13.8
morf 0.001	15.7	13.5	13.1
morf 0.01	15.8	14.7	14.2
morf 0.1	15.7	14.2	14.0
word	16.7	15.0	14.6

(d) Comparison of different language models using a TDNN-BLSTM acoustic model and the +m+-subword marking style.

We use 1500 hours of acoustic modeling data from three different data sets. The Speecon corpus [17] contains read speech in multiple different conditions. The Speechdat database [32] also contains read speech from a high number of speakers over telephone lines. Lastly, the parliament corpus [21] is used which has speech from the Finnish parliament. For evaluation, we use a broadcast news set, obtained from the Finnish national broadcaster YLE. The test conditions are the same as in [36].

For language modeling we used data from the Finnish Text Collection [10] which contains mainly newspapers and books from around the turn of the century. Our selection contained 12M sentences with in total 143M tokens. The number of unique words was 4.2M.

Based on the initial experiments, where we trained three different acoustic model architectures and a small word-based n-gram language model, we chose the TDNN-BLSTM acoustic model for the Finnish task (see Table 3a). Note that later experiments show that the optimal acoustic model depends on the type of language model used and that in some cases a TDNN model could perform better (see Table 10a).

For Finnish, we trained two n-gram models for all segmentations and all different marking styles. The first-pass n-gram models were tuned to have appr. 4M n-gram-contexts. For the rescored models the amount of n-gram contexts depends on the level of segmentation, ranging from 50M contexts for the word model to 200M contexts for the character models. Table 3b shows that the +m+-style marker is giving the best performance, which is in line with [37,36]. The Morfessor-based subword models are giving the best performance, with the best segmentation having a 5% relative improvement over a word-based model. The character model is performing similarly to the word model, having only a 1% relative degradation.

When we compare the speech recognition results with the perplexity values on the same development set (Table 3c), we see that they do not follow the same pattern. We calculated the perplexity values also for the other languages and observed similarly that they do not indicate the best subword segmentation and marking style for speech recognition. For example, the +m+-style tends to perform better in speech recognition than in language modeling, as it can help to disambiguate pronunciation variants for the same grapheme sequences.

When we rescore and interpolate the results with RNN-based language models we see a remarkable improvement for all language models. As expected, the character-based model benefits most from RNNLMs, probably because the RNN is more effective than n-gram in capturing long contexts. When the smaller 'shallow' network and the larger 'deep' network are compared, the improvement is largest for the character-based model. Compared to the previous best result in [36], 14.0%, we outperform it by 6.5% relative.

3.3 Arabic

The Arabic, as well as the Finnish language, has a structure that makes it naturally suitable for subword-based speech recognition. In previous work, linguistic units have been used frequently, but also data-driven units have been applied successfully [7,8,26].

Both the acoustic modeling and language modeling data used in this work come from the 2016 MGB-challenge. The audio is multi-genre broadcast data from Al-Jazeera and the language modeling text has been sourced from transcripts and the Al-Jazeera website. In total 1020 hours of data is used for

Table 3 Evaluations for the Arabic MGB-2 broadcast data set

TDNN	TDNN-LSTM	TDNN-BLSTM
20.8	19.4	**18.2**

(a) Comparison of different acoustic models using a word n-gram language model.

Segmentation	Vocab size	<w>	+m+	m+	+m
char	40	17.8	18.1	18.3	18.3
morf 0.001	5406	17.3	17.2	17.2	17.4
morf 0.01	28802	17.4	17.2	17.1	17.4
morf 0.1	111713	17.5	17.3	17.3	17.5
word	1303163	17.7			

(b) Rescored n-gram results for different subword markers and segmentations using a TDNN-BLSTM acoustic model..

		RNNLM	
Segmentation	n-gram	shallow	deep
char	18.3	16.8	16.5
morf 0.001	17.2	16.0	15.7
morf 0.01	17.1	15.8	15.6
morf 0.1	17.3	16.4	16.2
word	17.7	16.7	16.5

(c) Comparison of different language models using a TDNN-BLSTM acoustic model.

acoustic model training and 121M tokens of text for language model training. The evaluation set used is the development set provided for the MGB2 challenge [1].

Although previously phoneme-based lexicons have shown better performance than grapheme-based lexicons [2], we have opted to use only grapheme-based lexicons to be able to run all subword systems with the same acoustic model and without preparing pronunciation dictionaries for the subwords. Note that our grapheme-based system for the MGB-3 challenge outperformed all competitors, even when only word units were used [35, 3].

As for Finnish, we tested again all TDNN, TDNN-LSTM, and TDNN-BLSTM acoustic models. Table 4a shows that for Arabic the TDNN-BLSTM outperforms the other models by a considerable margin, hence TDNN-BLSTM models are used for further experiments.

Table 4b shows the results for different segmentations and marking styles. Unlike for Finnish, none of the markings outperform the others clearly. As the best result (17.1%) was obtained with the m+-marking, we use this in further experiments. The different segmentations perform in a similar way as

Table 4 Evaluations for the Swedish Språkbanken read speech data set

TDNN	TDNN-LSTM	TDNN-BLSTM
6.3	**6.1**	6.9

(a) Comparison of different acoustic models using a word n-gram language model.

Segmentation	Vocab size	<w>	+m+	m+	+m
char	32	3.9	3.3	3.3	3.7
morf 0.001	10996	4.1	3.1	3.3	3.3
morf 0.01	50537	4.3	3.1	3.2	3.2
morf 0.1	197022	4.5	3.0	3.2	3.2
word	3543864	3.0			

(b) Rescored n-gram results for different subword markers and segmentations using a TDNN-LSTM acoustic model.

		RNNLM	
Segmentation	n-gram	shallow	deep
char	3.3	2.8	2.5
morf 0.001	3.1	2.4	2.3
morf 0.01	3.1	2.7	2.5
morf 0.1	3.0	2.6	2.4
word	3.0	2.7	2.5

(c) Comparison of different language models using a TDNN-LSTM acoustic model.

the Finnish ones, with the Morfessor-based models outperforming both word and character-based models.

After the RNN-based rescoring, the Morfessor segmentations are still performing the best, with the *morph 0.01* segmentation achieving a result of 15.6% word error rate. Again, the character-based model has the highest gain from increasing the depth and complexity of the RNN model. The previous best, single-system result, was 15.9% [35].

3.4 Swedish

Swedish is a North Germanic language in the Indo-European language family. As these languages do not typically have very phonemic orthography, subword-based ASR models are not common. However, Swedish has many compound words, which suggests that subword units could work well for the lexicon.

We train our models with the data provided by the Språkbanken corpus, a public domain corpus hosted by the National Library of Norway. We used 354 hours of acoustic data from the training section of the corpus for training and

9 hours of acoustic data for development and evaluation, which is roughly 50% of the provided evaluation data.

For language modeling purposes the Språkbanken corpus contains n-gram-counts up to the 6th order, calculated from their language modeling texts. Unfortunately, the source texts, required for doing subword n-gram or RNN-based modeling, are not available. To overcome this, we reconstructed an approximation of the original language modeling corpus from the n-gram-counts that is 6-count-consistent. The procedure for this was simple, by starting with a 6-gram context that begins with a sentence marker and then obtaining the next word by finding a new 6-gram that is consistent with the last 5-gram of the current sentence. This repeats until a sentence-end marker is found and a new sentence will be started. All counts for 6-grams are decreases whenever the 6-gram is used, resulting in all 6-grams being used exactly the same amount of times as in the original language modeling corpus. Our reconstructed language modeling corpus has 398M tokens.

Although Swedish is semi-phonemic, there are enough deviations that normally a phoneme-based lexicon would be useful. Thus, the choice of implementing grapheme-based models might affect the performance. Although we have not found a comparison between phonemic and grapheme-based lexicons for Swedish, we assume based on the already excellent results showed later in this section (Table 5c) that a phoneme-based lexicon would not improve this system much further.

Table 5a shows that the baseline for this data set already achieves a very low word error rate. In contrast to the Finnish and Arabic systems, the TDNN-LSTM system outperforms the TDNN-BLSTM system. Altough we have not investigated this in detail, we expect that the reason is the smaller amount of acoustic data in the corpus. The optimal TDNN-LSTM model had 26M parameters, while the optimal TDNN-BLSTM had already 45M parameters.

Like in the Finnish task, Table 5b shows that the +m+-marking style performs best for Swedish, with the <w>-style being far inferior over other styles. In n-gram-based modeling the word model slightly outperforms the Morfessor and character-based models.

Even though the word error rates for n-gram-based models are already very low, Table 5c shows that RNN-based language models still improve those results by a significant margin. Even more surprising is that character-based models can match the performance of word-based models. To test the diversity between these two models, we calculate the cross word error rate (cWER) [43]. Between the 'deep' RNN word and character model the cWER is 1.63%, showing that the majority of the errors in the models are not shared. This suggests an excellent opportunity for using these to models in an ensemble or system combination.

Table 5 *Grapheme vs. Phonemes for English using word-based language models.*

	Grapheme	Phoneme
n-gram small	21.4	20.5
n-gram rescore	18.9	17.9
RNN 'deep'	17.7	16.6

3.5 English

Lastly, we run the same set of experiments on an English data set. Unlike the previous languages, English has only a very limited number of inflections and surface variations for each word form, and there is a weak relation between the surface form and the pronunciation of a word. Traditionally, phoneme-based systems have always been far superior to grapheme-based system. However, it has been shown that for the 2018 MGB English data set that we are using, there is only an appr. 5% relative improvement of phoneme-based systems over grapheme-based systems [42]. Therefore, even though we do not expect any necessary improvement of subword-based models over word-based models, we decided to create a grapheme-based system and evaluate the performance of word, subword and character units also in this task.

In the conditions of the official 2018 MGB challenge, the segmentation of the hour-long broadcasts needs to be done automatically. Unfortunately, we did not have a segmenter available and instead, we use the utterance level segment timings provided in the reference file. We do expect a small degradation when preparing the official results using a segmentation algorithm trained on the challenge data. In total, we used 283 hours of acoustic training data and a text corpus that consists of 646M tokens. For development and evaluation, we use the *dev17b* development set.

To validate the findings of [42], we train both a grapheme and phoneme-based TDNN acoustic model for English, and compare three different word language models. Table 5 shows that in our setup the results are similar. For a simple n-gram language model, we have only a 4.3% relative WER degradation for a grapheme-based model and for the most advanced model a 6.7% relative reduction. The reason why the difference between these models is so small is not entirely evident, but we suspect that the modeling power of the acoustic models in combination with the lattice-free MMI criterion is able to compensate for the loose grapheme-phoneme relationship. However, this would require further research to be validated. As the results are very close, we are strengthened in the belief that it is possible to make competitive models for English without the use of a hand-crafted phoneme lexicon.

Table 7a shows that here the TDNN (with 18M parameters) outperforms all other architectures. Given that we had the least amount of training data for this language this is not surprising and we expect that with more data to train the recurrent architectures properly they would be able to cover the wider pronunciation variation even better.

Table 6 Evaluations for the English MGB dev17b broadcast data set

TDNN	TDNN-LSTM	TDNN-BLSTM
21.4	21.6	24.2

(a) Comparison of different acoustic models using a word n-gram language model.

Segmentation	Vocab size	<w>	+m+	m+	+m
char	29	20.6	20.0	20.0	20.4
morf 0.001	9712	19.4	18.7	18.8	18.9
morf 0.01	35441	19.6	18.8	18.8	18.8
morf 0.1	100983	19.7	18.8	18.9	18.8
word	757627	18.9			

(b) Rescored n-gram results for different subword markers and segmentations using a TDNN acoustic model.

		RNNLM	
Segmentation	n-gram	shallow	deep
char	20.0	18.6	18.1
morf 0.001	18.7	17.9	17.8
morf 0.01	18.8	18.1	17.5
morf 0.1	18.8	17.4	17.3
word	18.9	18.3	17.7

(c) Comparison of different language models using a TDNN acoustic model.

Table 7b shows that for n-gram modeling the Morfessor-based models slightly outperform the word model. This is a surprising result, as previous attempts to make subword models for English have not had great success. Like the most other languages, the +m+-style marker performs best and is chosen for further experiments. After rescoring with an RNN-based LM we see that the subword models still outperform the word models. Table 7c also shows that whereas for n-gram models there is still a 1.1% absolute gap between word and character-based models, the gap is only 0.4% when using the deep RNNLMs.

To analyze the consistency of the English results, we have split out the test-set into the genres given by the creators of the MGB-challenge. Table 7 shows that for all categories, except the ‘advice’ genre, the Morfessor subword models outperform the word-based models.

3.6 Discussion

Looking for the common patterns in the results between languages, we first notice that the Swedish results are much better than any other tasks. A natural explanation is that it consists of read speech in a controlled environment, which

Table 7 Word error rates for different English categories using the RNN 'deep' language model

	Total	category					
		advice	childrens	comedy	documentary	drama	events
% of total		14.3	11.3	23.1	35.3	6.1	10.0
char	18.1	14.5	16.9	25.7	14.5	14.9	22.1
morf 0.1	17.4	14.5	**16.0**	**24.4**	**13.8**	**14.6**	**21.5**
word	17.7	**14.4**	16.1	25.0	14.1	14.8	22.0

Table 8 Amount of data used for the different languages

	Finnish	Arabic	Swedish	English
Speech (training)	1500h	1020h	354h	283h
Speech (evaluation)	5.4h	8.4h	8.7h	5.7h
Text (training tokens)	143M	121M	398M	646M
Unique tokens	4.3M	1.3M	3.5M	0.8M

is in contrast with all other data sets that have been taken from TV-broadcasts with both spontaneous speech and background noises.

Among different type of acoustic models, only those languages that had lots of data available ($>$ 1000 hours) seem to be able to take advantage of TDNN-BLSTM models. If there are less data, the TDNN-BLSTMs do not achieve the same performance as TDNN or TDNN-LSTM models. Note that the number of parameters was (coarsly) optimized for each single acoustic model such that reducing or increasing the number of parameters did not improve the results.

The differences between word and Morfessor-based subword models follow a common pattern, where the subword models outperform the corresponding word-based models. The only exception to this are the n-gram models for Swedish which are slightly worse or equal to the word-based result. For RNN-based language models, the subword units show more significant improvements, and the difference between the best subword and word model is more substantial than for the n-gram models.

For Finnish and Arabic it has been shown before that subword models can outperform word-based models. Our results validate that this is still the case in a modern WFST-based HMM/DNN speech recognizer, especially when RNN-based language models are used. For Swedish and English this is, as far as we are aware, the first time that Morfessor-subword models outperform a word-based model on a large vocabulary system trained on a large language modeling corpus. While the single best result for English, 17.3% WER, is worse than the best result for a phoneme-based system (16.6%, see Table 5), the system combination of various grapheme-based models improve the results significantly. This will be discussed in Section 4.

When comparing the character-based results to the word-based results we see an improvement over word-based models only for Finnish (14.6% vs 13.8%). In Arabic and Swedish the RNN-based character models match the corresponding word-models (16.5% and 2.5% resp.) and for English there is a small degradation (17.7% vs 18.1%). These results are still impressive, as they were made with much smaller RNN models (because of the reduced size of the in- and ouput layers) that took less time to train, especially when the time needed to cluster the vocabulary of a word-based model is taken into account. Another thing to note about the character-based models is that they had the largest relative improvement between n-gram and RNN-based models, as well as the most significant relative improvement between 'shallow' and 'deep' RNN models. This indicates that further refinement of the RNN-based language models might lead to even better results, possibly outperforming word-based models for all languages.

The Morfessor-based subword models did outperform the character models for all languages, indicating that the subword units learned by Morfessor are very suitable for the ASR task. The optimal number of subword units varied per language, with appr. 10k units being optimal for Finnish and appr. 100k units for English. However, the differences were small, and we expect that for any language a subword lexicon with a size between 10k and 100k will work well and provide accuracy that is close to the optimal lexicon.

4 System combination between different units

Introducing different sizes of subword units gives a possibility to make a variety of models to evaluate. Not only can these models be evaluated separately, but it is also possible to combine these results using system combination. The most common technique is to use Minimum Bayes Risk (MBR) decoding [13, 44]. Instead of optimizing the sentence-error-rate, which is done in ordinary Maximum a Posterori (MAP) decoding, MBR decodes the utterance in such way that minimizes the word error rate directly. This is very powerful in combination with lattice combination, as for each word location the language model likelihood can be averaged between the lattices.

To make system combination work between systems trained on different units, the lattices need to be first converted to the same vocabulary. This can be done with a simple FST-based transformation that is created by first reconstructing all the words in the subword lattice and mapping these subwords to their actual words. Once all lattices are using the same vocabulary, they can be combined into a new lattice that contains all word-paths from the different systems. Afterwards, a standard MBR-decoder is used to get the optimal word sequence.

For all languages evaluated in this paper, we have run system combination experiments in two dimensions. First, we run system combination separately for

Table 9 Each of the rows "char, morf, word" contain WER results of one LM trained as the RNN 'deep' LM introduced in the chapter III. The four columns are the three AMs and a combination of the corresponding systems (AM comb). The number in parenthesis is the number of systems combined. The LM/AM Comb rows are system combinations of all these LM units, as well as a selection of different AMs. The colored background indicates which of the AMs (columns) are combined.

Unit	TDNN	TDNN+LSTM	TDNN+BLSTM	AM comb
char	13.4	13.6	13.8	12.3 (3)
morf 0.001	12.8	13.3	13.1	11.9 (3)
morf 0.01	14.1	14.5	14.2	12.7 (3)
morf 0.1	13.8	14.3	14.0	12.4 (3)
word	14.6	15.0	14.6	13.3 (3)
LM/AM Comb			▨ 12.8 (5)	
	▨ 12.4 (5)			
		▨ 11.8 (10)	▨	
	▨	▨ 11.4 (15)	▨	

(a) Finnish

Unit	TDNN	TDNN+LSTM	TDNN+BLSTM	AM comb
char	18.9	17.8	16.5	15.9 (3)
morf 0.001	17.7	16.7	15.7	14.9 (3)
morf 0.01	17.7	16.6	15.6	14.9 (3)
morf 0.1	18.4	17.1	16.2	15.4 (3)
word	18.7	17.6	16.5	15.7 (3)
LM/AM Comb			▨ 15.5 (5)	
		▨ 14.7 (10)	▨	
	▨	▨ 14.6 (15)	▨	

(b) Arabic

Unit	TDNN	TDNN+LSTM	TDNN+BLSTM	AM comb
char	2.4	2.5	3.2	2.2 (3)
morf 0.001	2.2	2.3	3.1	2.1 (3)
morf 0.01	2.4	2.5	3.1	2.2 (3)
morf 0.1	2.3	2.4	2.9	2.1 (3)
word	2.4	2.5	3.1	2.2 (3)
LM/AM Comb		▨ 2.2 (5)		
	▨ 2.0 (10)	▨		
	▨	▨ 2.0 (15)	▨	

(c) Swedish

Unit	Phone TDNN	TDNN	TDNN+LSTM	TDNN+BLSTM	AM comb
char		18.1	18.7	21.2	16.9 (3)
morf 0.001		17.8	18.3	21.0	16.6 (3)
morf 0.01		17.5	17.9	20.7	16.4 (3)
morf 0.1		17.3	17.8	20.6	16.2 (3)
word	16.6	17.7	18.2	20.9	16.5 (3)
		▨ 17.2 (5)			

each language model unit, combining over the three different acoustic models we trained earlier. This shows the improvement for the typical setup where the system combination is used over multiple acoustic models. The second experiment we did was to combine the RNN-deep models for all different units: characters, Morfessor subwords, and words. In addition to doing this for a single acoustic model, we also combined all these units across the two best and all acoustic models.

Table 10a shows the system combination results for the Finnish models. The rightmost column, which shows the results of combinations across acoustic models, indicates that for any unit a significant gain can be made by combining the different AMs. The sharpest decrease in WER is obtained for the *morf 0.01* models, for which the best non-combined result 14.1% is reduced to 12.7% with system combination.

The last rows of Table 10a shows the results of system combination over different units. The first result, which combines the 5 TDNN+BLSTM systems improves the best individual result of 13.1% to 12.8%. A combination over multiple acoustic models enhances the result even further to 11.4%. Note that the TDNN model performs best when using RNNLMs, in contrast to TDNN+BLSTM for n-gram-based models (Table 3a).

For Arabic, the results in Table 10b show a similar improvement through system combination. The best cross-AM combination improved the best individual result of 15.7% to 14.9% and combination with multiple units gives the best result of 14.6%. These results are in line with the improvements we obtained in [35] where the combination of all our systems improved the best individual result of 15.9% to 14.8%. The main reason for obtaining better results than [35] is the use of better (deeper) RNNLMs.

For Swedish, the best WER for an individual system was already very low 2.2%. However, the system combination over different acoustic models reduced this to 2.1% and the combination over both acoustic models and language units to 2.0% WER.

Lastly, the English results are more extensive, as we combined the systems also with the phoneme-based acoustic word model. The best individual result with the phoneme-based model is 16.6% and with the grapheme-based model 17.3% WER. Already when three different grapheme-based acoustic models are combined, the error rate matches or surpasses that of the single phoneme-based system in for all units, except for the character-based model. Combining all systems gives the best WER of 15.9%, a 8.0% relative improvement over the best single grapheme-based system and a 4.2% relative improvement over the phoneme-based system. Note that this comparison only indicates the power of combining diverse models, in larger extent across acoustic models and in smaller extent across different units. It also would be possible—and probably effective—to do system combinations across different acoustic models for phoneme-based models. However, for a phoneme-based system it is much more difficult to train models with different kind of units to be used for system combinations From the perspective of low-resource languages that do not have manually created lexicon resources, it is encouraging that even for a language with as irregular pronunciations as English, a normal phoneme-based system can be matched by multiple grapheme-based systems.

To further optimize the recognition accuracy we could also try system combination over different phoneme-based acoustic models as we did for grapheme-based models. However, with the current experiments we can already conclude that our grapheme-based systems are able to capture the different pronunciations of graphemes with surprising accuracy. To our knowledge, it has not been shown before that the difference between grapheme and phoneme-based systems can be this small for the English language.

5 Under-resourced scenario

As explained in Section 2.3, one of the strengths of subword models, including character models, is that the language model training data contains more examples of each unit and therefore can be more effectively used in language model training compared to the word models. To demonstrate this, we trained all systems with only 10% of the available language modeling data and compared the impact on the performance for different language modeling units. This effectively simulates an under-resourced scenario where less language modeling data is available.

In this study we reduced only the amount of language modeling data and kept the acoustic models identical to the previous experiments. Naturally, the acoustic model will also learn some (grapheme-based) language patterns through the sequence-based training criterion, but we do not expect this to affect the experiment: The language modeling data does not contain the acoustic model training utterances, and the number of word types and tokens in the acoustic modeling data is limited. In the Finnish experiment, where we have the largest acoustic modeling training set, there are still only 9M tokens and 400k unique words present in the acoustic training data. Even if this data were counted into the amount of language modeling data, the total amount of data used would still be less than 20% of the original text data.

For Finnish (Table 11a), the results are between 7 and 17% worse than with the full language modeling data. As hypothesized, the results of the subword models degrade less than those of the word-based models. The character model is now outperforming both the word model as well as two of the Morfessor subword models.

Table 11b shows a similar pattern for the Arabic data set. The character model outperforms the word-based model even for the n-gram model.

In Swedish, as shown in Table 11c, the relative degradations seem larger than for the other tasks. This may be caused by the nature of errors when the WER is much closer to zero, compared to the other tasks. In the under-resourced case, the character model outperforms the word-based model and comes close to the performance of the Morfessor-based subword models.

Unlike in the other languages, for English the performance of the subword models stays close to the word models both in the 10% and in the full 100%

Table 10 Comparison of different type of subword language models with 10% language modeling data. The last two columns show the deep RNNLM result for the full data and its relative difference to the deep RNNLM result with 10% data.

		RNNLM		100%	rel. diff
Segmentation	n-gram	shallow	deep	deep	to 10%
char	18.5	15.9	15.4	13.8	-10.4%
morf 0.001	17.4	15.4	15.4	13.1	-14.9%
morf 0.01	17.5	16.0	15.7	14.2	-9.5%
morf 0.1	17.6	15.3	15.1	14.0	-7.3%
word	19.6	18.0	17.5	14.6	-16.6%

(a) Finnish

		RNNLM		100%	rel. diff
Segmentation	n-gram	shallow	deep	deep	to 10%
char	19.3	17.7	17.5	16.5	-5.7%
morf 0.001	18.1	17.1	17.0	15.7	-7.7%
morf 0.01	18.1	17.1	17.0	15.6	-8.2%
morf 0.1	18.5	17.2	17.0	16.2	-4.7%
word	19.4	19.4	18.3	16.5	-9.8%

(b) Arabic

		RNNLM		100%	rel. diff
Segmentation	n-gram	shallow	deep	deep	to 10%
char	6.0	4.2	3.7	2.5	-32.4%
morf 0.001	5.3	3.4	3.1	2.3	-25.8%
morf 0.01	5.3	3.8	3.5	2.5	-28.6%
morf 0.1	5.4	3.6	3.4	2.4	-29.4%
word	6.0	4.7	4.4	2.5	-43.2%

(c) Swedish

		RNNLM		100%	rel. diff
Segmentation	n-gram	shallow	deep	deep	to 10%
char	21.5	19.8	19.1	18.1	-5.2%
morf 0.001	20.6	18.4	18.2	17.8	-2.2%
morf 0.01	20.4	19.0	18.7	17.5	-6.4%
morf 0.1	20.4	18.6	18.5	17.3	-6.5%
word	22.0	19.3	18.8	17.7	-5.9%

(d) English

data. On the other hand, the amount of language modeling data for English was the largest (see Table 8), with the 10% system still containing 65M training tokens.

For all languages, there is a clear degradation in results when the amount of language modeling data is reduced. As expected, the subword models are more robustly handling the data sparsity in all languages. The only exception to this, English, can be explained by the fact that the original vocabulary is much smaller and the language modeling data much more substantial than for the other languages. When comparing the character-based models to the Morfessor subword models, only in Finnish their performance matches to two out of three Morfessor models, with the best Morfessor model still outperforming the character based model (15.1% vs 15.4%). In other languages, the character models still stay behind.

6 Conclusion

We set out to implement and evaluate the use of subword units in state-of-the-art speech recognition, including advanced neural network based acoustic and language models. To do this, we have evaluated word and subword systems for four distinct languages from different language families.

Our primary evaluation shows that models based on subwords derived by Morfessor consistently outperform word-based models on all tested languages. The optimal size of the subword lexicon varied across languages, from less than 10,000 in Finnish to a bit over 100,000 in English, but in no situation did subword models perform worse than the same model with word units. Although this effect is already present when using n-gram language models, it is even stronger for RNN-based language models, which have a better capability of capturing longer contexts.

With accurate RNN language models, using single characters as subword units also yields surprisingly good results. A character-based model outperformed the word-based model for Finnish, produced a similar performance for Arabic and Swedish, and underperformed the word-based model only for English.

As using different subword units provides with a variety of different models, we combined these systems using MBR-based system combination. Although it was already effective to combine acoustic models from three different architectures, using models with different language modeling units was also a success, with combination systems reducing the the WER of the single best system with over 10% relative.

Although it was not the original intent of this paper, we did obtain interesting results regarding the use of grapheme models for English. Not only was a simple direct word-based comparison between grapheme and phoneme models only showing a 6% edge for phoneme-based models, by doing system combination the results of the grapheme-based system could match and surpass the phoneme-based result, without having the need for a hand-crafted pronuciation lexicon. Furthermore, a combination with all the grapheme and phoneme-based systems that we trained resulted in a 15.9% WER for the MGB dev17b test set, which is by far the best result published on this data set.

Lastly, the evaluation with smaller language modeling data sets showed another strength of the subwords, where the degradation was significantly smaller than that of the word units. This confirms the hypothesis that using subword units reduces data sparsity and increases model robustness.

Acknowledgements Computational resources were provided by the Aalto Science-IT project and the CSC – IT Center for Science, Finland. This work was partially funded by the DigiTala project, Business Finland's Challenge Finland project TELLme and the Kone foundation. In addition, this work has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 780069.

References

1. Ali, A., Bell, P., Glass, J., Messaoui, Y., Mubarak, H., Renals, S., Zhang, Y.: The MGB-2 challenge: Arabic multi-dialect broadcast media recognition. In: SLT 2016 – IEEE Spoken Language Technology Workshop, pp. 279–284 (2016). DOI 10.1109/SLT.2016.7846277
2. Ali, A., Mubarak, H., Vogel, S.: Advances in dialectal Arabic speech recognition: A study using twitter to improve Egyptian ASR. In: IWSLT 2014 – International Workshop on Spoken Language Translation (2014)
3. Ali, A., Vogel, S., Renals, S.: Speech recognition challenge in the wild: Arabic MGB-3. In: ASRU 2017 – IEEE Workshop on Automatic Speech Recognition & Understanding (2017)
4. Arisoy, E., Can, D., Parlak, S., Sak, H., Saraclar, M.: Turkish broadcast news transcription and retrieval. IEEE Transactions on Audio, Speech, and Language Processing **17**(5), 874–883 (2009). DOI 10.1109/TASL.2008.2012313
5. Bisani, M., Ney, H.: Open vocabulary speech recognition with flat hybrid models. In: INTERSPEECH 2005 – Eurospeech, 9th European Conference on Speech Communication and Technology, pp. 725–728. Lisbon, Portugal (2005)
6. Botros, R., Irie, K., Sundermeyer, M., Ney, H.: On efficient training of word classes and their application to recurrent neural network language models. In: INTERSPEECH 2015 – 16th Annual Conference of the International Speech Communication Association, pp. 1443–1447. Dresden, Germany (2015)
7. Choueiter, G., Povey, D., Chen, S.F., Zweig, G.: Morpheme-based language modeling for Arabic LVCSR. In: ICASSP 2006 – IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 1053–1056 (2006). DOI 10.1109/ICASSP.2006.1660205
8. Creutz, M., Hirsimäki, T., Kurimo, M., Puurula, A., Pylkkönen, J., Siivola, V., Varjokallio, M., Arisoy, E., Saraçlar, M., Stolcke, A.: Morph-based speech recognition and modeling of out-of-vocabulary words across languages. ACM Transactions on Speech and Language Processing **5**(1), 3:1–3:29 (2007). DOI 10.1145/1322391.1322394
9. Creutz, M., Lagus, K.: Induction of a simple morphology for highly-inflecting languages. In: Proceedings of the Seventh Meeting of the ACL Special Interest Group in Computational Phonology, pp. 43–51. Association for Computational Linguistics, Barcelona, Spain (2004)
10. CSC - IT Center for Science: The Helsinki Korp Version of the Finnish Text Collection (1998). URL http://urn.fi/urn:nbn:fi:lb-2016050207
11. Enarvi, S., Kurimo, M.: TheanoLM — an extensible toolkit for neural network language modeling. In: INTERSPEECH 2016 – 17th Annual Conference of the International Speech Communication Association, pp. 3052–3056. San Francisco (2016). DOI 10.21437/Interspeech.2016-618
12. Enarvi, S., Smit, P., Virpioja, S., Kurimo, M.: Automatic speech recognition with very large conversational Finnish and Estonian vocabularies. IEEE/ACM Transactions on Audio, Speech, and Language Processing **25**(11), 2085–2097 (2017). DOI 10.1109/TASLP.2017.2743344
13. Goel, V., Byrne, W.J.: Minimum Bayes-risk automatic speech recognition. Computer Speech & Language **14**(2), 115 – 135 (2000). DOI https://doi.org/10.1006/csla.2000.0138

14. Hirsimäki, T., Creutz, M., Siivola, V., Kurimo, M., Virpioja, S., Pylkkönen, J.: Unlimited vocabulary speech recognition with morph language models applied to Finnish. Computer Speech & Language **20**(4), 515–541 (2006). DOI 10.1016/j.csl.2005.07.002
15. Hirsimäki, T., Pylkkönen, J., Kurimo, M.: Importance of high-order n-gram models in morph-based speech recognition. IEEE Transactions on Audio, Speech, and Language Processing **17**(4), 724–732 (2009). DOI 10.1109/TASL.2008.2012323
16. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation **9**(8), 1735–1780 (1997). DOI 10.1162/neco.1997.9.8.1735
17. Iskra, D.J., Grosskopf, B., Marasek, K., van den Heuvel, H., Diehl, F., Kiessling, A.: SPEECON-Speech databases for consumer devices: Database specification and validation. In: LREC (2002)
18. Kirchhoff, K., Vergyri, D., Bilmes, J., Duh, K., Stolcke, A.: Morphology-based language modeling for conversational Arabic speech recognition. Computer Speech & Language **20**(4), 589 – 608 (2006). DOI http://dx.doi.org/10.1016/j.csl.2005.10.001
19. Kneser, R., Ney, H.: Forming word classes by statistical clustering for statistical language modelling. In: R. Köhler, B.B. Rieger (eds.) Contributions to Quantitative Linguistics, pp. 221–226. Kluwer Academic Publishers, Dordrecht, the Netherlands (1993). DOI 10.1007/978-94-011-1769-2_15
20. Kuo, H.K., Arısoy, E., Emami, A., Vozila, P.: Large scale hierarchical neural network language models. In: INTERSPEECH 2012 – 13th Annual Conference of the International Speech Communication Association, pp. 1672–1675. Portland, OR, USA (2012)
21. Mansikkaniemi, A., Smit, P., Kurimo, M.: Automatic construction of the Finnish Parliament Speech Corpus. In: INTERSPEECH 2017 – 18th Annual Conference of the International Speech Communication Association. Stockholm, Sweden (2017)
22. Mikolov, T., Karafit, M., Burget, L., Cernocký, J., Khudanpur, S.: Recurrent neural network based language model. In: INTERSPEECH 2010 – 11th Annual Conference of the International Speech Communication Association, pp. 1045–1048. Makuhari, Japan (2010)
23. Mirjam Killer Sebastian Stuker, T.S.: Grapheme based speech recognition. In: INTERSPEECH 2003 – Eurospeech, 8th European Conference on Speech Communication and Technology. Geneva, Switzerland (2003)
24. Mohri, M., Pereira, F., Riley, M.: Speech recognition with weighted finite-state transducers. In: Springer Handbook of Speech Processing, pp. 559–584. Springer (2008)
25. Morin, F., Bengio, Y.: Hierarchical probabilistic neural network language model. In: Aistats, vol. 5, pp. 246–252. Citeseer (2005)
26. Mousa, A.E.D., Kuo, H.K.J., Mangu, L., Soltau, H.: Morpheme-based feature-rich language models using deep neural networks for LVCSR of Egyptian Arabic. In: ICASSP 2013 – IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 8435–8439 (2013). DOI 10.1109/ICASSP.2013.6639311
27. Peddinti, V., Povey, D., Khudanpur, S.: A time delay neural network architecture for efficient modeling of long temporal contexts. In: INTERSPEECH 2015 – 16th Annual Conference of the International Speech Communication Association, pp. 3214–3218. Dresden, Germany (2015)
28. Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., Hannemann, M., Motlicek, P., Qian, Y., Schwarz, P., Silovsky, J., Stemmer, G., Vesely, K.: The Kaldi speech recognition toolkit. In: ASRU 2011 – IEEE Workshop on Automatic Speech Recognition & Understanding (2011)
29. Povey, D., Peddinti, V., Galvez, D., Ghahremani, P., Manohar, V., Na, X., Wang, Y., Khudanpur, S.: Purely sequence-trained neural networks for ASR based on lattice-free MMI. In: INTERSPEECH 2016 – 17th Annual Conference of the International Speech Communication Association, pp. 2751–2755. San Francisco (2016). DOI 10.21437/Interspeech.2016-595
30. Prabhavalkar, R., Rao, K., Sainath, T.N., Li, B., Johnson, L., Jaitly, N.: A comparison of sequence-to-sequence models for speech recognition. In: INTERSPEECH 2017 – 18th Annual Conference of the International Speech Communication Association, pp. 939–943. Stockholm, Sweden (2017). DOI 10.21437/Interspeech.2017-233
31. Rao, K., Sak, H.: Multi-accent speech recognition with hierarchical grapheme based models. In: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4815–4819 (2017). DOI 10.1109/ICASSP.2017.7953071

32. Rosti, A., Rämö, A., Saarelainen, T., Yli-Hietanen, J.: Speechdat Finnish database for the fixed telephone network. Tech. rep., Tampere University of Technology (1998)
33. Sainath, T.N., Prabhavalkar, R., Kumar, S., Lee, S., Kannan, A., Rybach, D., Schogol, V., Nguyen, P., Li, B., Wu, Y., Chen, Z., Chiu, C.: No need for a lexicon? evaluating the value of the pronunciation lexica in end-to-end models. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5859–5863 (2018). DOI 10.1109/ICASSP.2018.8462380
34. Siivola, V., Hirsimäki, T., Virpioja, S.: On growing and pruning Kneser-Ney smoothed n-gram models. IEEE Transactions on Audio, Speech & Language Processing **15**(5), 1617–1624 (2007)
35. Smit, P., Gangireddy, S.R., Enarvi, S., Virpioja, S., Kurimo, M.: Aalto system for the 2017 Arabic multi-genre broadcast challenge. In: ASRU 2017 – IEEE Workshop on Automatic Speech Recognition & Understanding (2017)
36. Smit, P., Gangireddy, S.R., Enarvi, S., Virpioja, S., Kurimo, M.: Character-based units for unlimited vocabulary continuous speech recognition. In: ASRU 2017 – IEEE Workshop on Automatic Speech Recognition & Understanding (2017)
37. Smit, P., Virpioja, S., Kurimo, M.: Improved subword modeling for WFST-based speech recognition. In: INTERSPEECH 2017 – 18$^{\text{th}}$ Annual Conference of the International Speech Communication Association. Stockholm, Sweden (2017)
38. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research **15**(1), 1929–1958 (2014)
39. Srivastava, R.K., Greff, K., Schmidhuber, J.: Training very deep networks. In: C. Cortes, N.D. Lawrence, D.D. Lee, M. Sugiyama, R. Garnett (eds.) Advances in Neural Information Processing Systems 28, pp. 2377–2385. Curran Associates, Inc. (2015)
40. Tarján, B., Fegyó, T., Mihajlik, P.: A bilingual study on the prediction of morph-based improvement. In: SLTU, pp. 131–138 (2014)
41. Virpioja, S., Smit, P., Grönroos, S.A., Kurimo, M.: Morfessor 2.0: Python implementation and extensions for Morfessor Baseline. Report 25/2013 in Aalto University publication series SCIENCE + TECHNOLOGY, Department of Signal Processing and Acoustics, Aalto University, Helsinki, Finland (2013)
42. Wang, Y., Chen, X., Gales, M., Ragni, A., Wong, J.: Phonetic and graphemic systems for multi-genre broadcast transcription. In: ICASSP 2018 – IEEE International Conference on Acoustics, Speech and Signal Processing (2018)
43. Wong, J.H.M., Gales, M.J.F.: Multi-task ensembles with teacher-student training. In: ASRU 2017 – IEEE Workshop on Automatic Speech Recognition & Understanding, pp. 84–90 (2017). DOI 10.1109/ASRU.2017.8268920
44. Xu, H., Povey, D., Mangu, L., Zhu, J.: An improved consensus-like method for minimum Bayes risk decoding and lattice combination. In: ICASSP 2010 – IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 4938–4941 (2010). DOI 10.1109/ICASSP.2010.5495100

A.5 AALTO's DCASE 2018 workshop paper [5]

This paper describes AALTO's submission to the DCASE 2018 general audio event detection challenge and the methods applied and developed for it. The detected tags, time codes and probabilities of the audio events will be used in the project together with visual tags and speech recognition results as inputs to the video description system.

THE AALTO SYSTEM BASED ON FINE-TUNED AUDIOSET FEATURES FOR DCASE 2018 TASK2 —— GENERAL PURPOSE AUDIO TAGGING

Zhicun Xu, Peter Smit, Mikko Kurimo

Aalto University, Department of Signal Processing and Acoustics, Espoo, Finland,
{zhicun.xu, peter.smit, mikko.kurimo}@aalto.fi

ABSTRACT

In this paper, we presented a neural network system for DCASE 2018 task 2, general purpose audio tagging. We fine-tuned the Google AudioSet feature generation model with different settings for the given 41 classes on top of a fully connected layer with 100 units. Then we used the fine-tuned models to generate 128 dimensional features for each 0.960s audio. We tried different neural network structures including LSTM and multi-level attention models. In our experiments, the multi-level attention model has shown its superiority over others. Truncating the silence parts, repeating and splitting the audio into the fixed length, pitch shifting augmentation, and mixup techniques have all improved the results by a reasonable amount. The proposed system achieved a result with MAP@3 score at 0.936, which outperforms the baseline result of 0.704 and achieves top 8% in the public leaderboard.

Index Terms— audio tagging, AudioSet, multi-level attention model

1. INTRODUCTION

Sound contains various information that could indicate the sound sources, surrounding environment, music genres, possible dangers or even the emotions of the speakers. Thus sound plays a crucial part in our daily communication and interaction with the world. Teaching machines to listen, such as recognizing the sound events, would benefit humans in many ways. Relevant applications include public surveillance, sound print, auditory medical information monitoring and multimedia content analysis.

General purpose audio tagging is a task that infers descriptive labeling from these sounds such as musical instruments, domestic animals, and human activities. Recognizing and labelling these sound events with appropriate tags can provide a powerful tool to categorize the extensively large amount of audio data from the internet. With the labels, the content providers can give better services such as providing audio descriptions for visually or hearing impaired people, and providing powerful searching tools for the people working in the entertainment industries.

The traditional methods for doing the audio classification and audio tagging are adapted from speech recognition such as the Mel-frequency cepstral coefficients (MFCC) features and simple Gaussian mixture model (GMM) classifiers [1]. Recently, deep neural networks have proven its great usefulness in feature engineering, classification, detection, and audio synthesis. Almost all the submissions in DCASE 2017 [2] used some forms of neural networks such as long short-term memory units (LSTM) and convolutional neural networks (CNN). Thus deep learning is our main research approach for this task. Google has created a dataset called AudioSet with a structured hierarchical ontology [3], which provides the proper way to annotate the sounds. Instead of releasing the original audio files, each sample from the AudioSet is represented by 10 instances of 128 dimensional features. Google also provides a pre-trained model to generate the 128 features for 0.960 seconds audio. Kong *et al.* proposed a single-level attention model on this dataset, which outperformed the Google' baseline [4]. Later, Yu *et al.* proposed a multi-level attention model as an extension to previous single-level attention models, the results outperformed both the single-level attention model and the baseline [5].

Thus we came up with the idea to fine-tune the feature generation model first for the 41 classes of this DCASE task, and then try the multi-level attention model on the generated compact features. We aimed at proving the usability of these compact features and the superiority of multi-level attention model. We also incorporated pitch shifting augmentation and mixup techniques.

The structure of this paper is as follows. Section 2 describes methods on how to fine-tune the existing CNN model for the given 41 classes. Section 3 describes the design of our proposed system. The experimental results and conclusions can be seen in section 4 and 5 respectively.

2. FINE-TUNED VGGISH MODEL

2.1. Structure

VGGNet [6] with deep CNN structures has worked greatly well in image classification. Since the spectral representation of an audio signal can be used directly as an image, this deep CNN structure is also a promising techniques in many machine listening tasks, such as audio tagging, audio event detection and acoustics scene classification. VGGish model [7] is a variant of the VGG model with minor modifications. Table 1 shows the detailed structure of the modified version. Google trained this model on the YouTube-100M dataset with a total of 100 million videos. The training set contains 70 million videos and they were further split into non-overlapping 960 ms audio frames. Log-mel spectrograms were then computed as 96$\times$64 images for the input of the VGGish model. The evaluation results of VGGish model showed its great usability in the audio domain.

The pre-trained VGGish model can act as a feature extractor. The provided 128 embedding features for 0.960 seconds are very compact, high level and semantically meaningful as well. These

This work was supported by the Kone foundation, and the European Union's Horizon 2020 research and innovation programme via the project MeMAD (GA780069). Computational resources were provided by the Aalto Science-IT project.

Table 1: VGGish model structure. The kernel size for CNN is (3, 3) and the kernel size for pooling is (2, 2). The activation function is 'relu' for all the layers.

layers	**filters@size**	**layers**	**filters@size**
1. input	1@96×64	**9. conv**	512@12×8
2. conv	64@96×64	**10. conv**	512@12×8
3. pooling	64@48×32	**11. pooling**	512@6×4
4. conv	128@48×32	**12. flatten**	12288
5. pooling	128@24×16	**13. fc**	4096
6. conv	256@24×16	**14. fc**	4096
7. conv	256@24×16	**15. fc**	128
8. pooling	256@12×8	**16. ...**	...

compact features can then be fed into a shallower model for classification. VGGish model can also become part of a larger model where more layers are added upon the model, which makes it possible to adapt and fine-tune this model for different datasets.

2.2. Balanced and Unbalanced Fine-tuning

The original VGGish model is trained for a multi-label classification task with the Google AudioSet ontology [3], which has 632 classes in a hierarchy tree structure. This challenge is a multi-class classification task, which means there is only one correct label for a sample. The 41 classes for this task all belongs to the AudioSet ontology. The training data is provided by FreeSound Dataset (FSD) [8]. The dataset has in total 9473 training files with the smallest class having only 94 training samples and the biggest class having 300 training samples. The average length of the audio files is 6.7 seconds. The more detailed description of the task setup, dataset and baseline can be seen in [9]. We added one hidden layer with 100 units after the VGGish model, and the final classification layer has 41 units with softmax activations.

To fine-tune the VGGish model, we must divide the data into a training part and a validation part. The validation loss is then used as the stopping criterion in case of overfitting. Firstly, we used the first 8000 audio files as training data and the remaining 1473 audio files as validation data. To fully use the data, we then used the last 8000 audio files as training data and the first 1473 audio files as validation data. Thus for one balancing technique we would get two models, one of which is fine-tuned on the first 8000 audio files and the other is fine-tuned on the last 8000 audio files.

Mini-batch balancing is a technique which assigns equal number of training samples for each class in each mini-batch. It has been proven useful on AudioSet to deal with the unbalanced dataset [4]. For the balanced fine-tuning, we followed the mini-batch balancing method by choosing an audio sample from each class for every training batch. However, it is not perfect balancing since audio samples may contain different number of 0.960 seconds length segments. In total, there would be 41 audio files, around 280 log-mel spectrograms, for each training batch. On the other hand, we also fine-tuned a unbalanced model by randomly choose 41 audio files in each batch for comparison. So we got 4 models in total.

The fine-tuned model are used as the feature extractors for the audio. Each audio file is firstly split into several non-overlapping 0.960 seconds segments. And for each segment, the log-mel spectrogram with dimension 96×64 is computed, then the spectrogram is fed into the fine-tuned model to get the 128 dimensional features.

3. THE PROPOSED SYSTEM

3.1. Preprocessing

The provided audio files are provided as PCM 16 bits, 44.1 kHz, mono format. However, the original quality might be quite different since they are uploaded by users all around the world. For preprocessing, we trimmed the silence parts only in the beginning and the end to avoid the influence of these irrelevant silence parts. We used the librosa[1] toolbox for the trimming here. The normalizing and pitch shifting mentioned in the following section are also implemented using this toolbox. The silence parts that are in the middle might contain important dynamic information for classification. We then normalized the amplitude of the audio files to [-1, 1]. The trimmed and normalized audio files are then split into non-overlapping 0.960 seconds segments. For each segment the log-mel spectrogram with size 96×64 is computed with 25ms window size and 10ms hop size. The 96 represent the frame size and 64 represents the number of frequency bands. The spectrograms are then fed into the fine-tuned VGGish model for features extraction.

3.2. Data Augmentation

The data is unbalanced where only around half of the classes have 300 audios. The imbalance problem could make the model emphasize more on the classes with more training samples and neglect to learn from the classes with less samples. To deal with this problem, we used the pitch shifting, repeat and split strategies for augmentation.

For the implementations of pitch shifting, we randomly choose an integer number between (-12, 12) for each audio file, and then shift the correspond number of semitones to create the new file. The shifting does not affect much on the melodic or stable classes, since the maximum shift is only one octave. For the noise-like classes such as 'fireworks' and 'fart', the perception is still relatively okay even for those with maximum amount of shift.

Some of our models used fixed length audio as input, but the audio files have variable length. To fully use the full length of the audio, we split them into several fixed length audios as input, which also gave us more training data. For the audio files that are shorter, we repeat them and concatenate them together to the fixed length.

3.3. Mixup

Mixup means training the neural networks using the convex combinations from pairs of examples and their labels. It helps the neural network to emphasize the linear combination between the training samples, which can improve the generalization ability, reduce memorization of the corrupted labels, increase the robustness to adversarial examples [10]. The training data provided by the organizer has another label representing if the audio is manually verified. Thus trying this method can help reduce the effect of the potential corrupted or misclassified audio in the unverified audios. Besides, this will also allow the model learn to distinguish between classes. The following equation [10] explains how the algorithms works:

$$\begin{aligned} \tilde{x} &= \lambda x_i + (1-\lambda) x_j \\ \tilde{y} &= \lambda y_i + (1-\lambda) y_j \\ \lambda &\sim \text{Beta}(\alpha, \alpha) \\ \alpha &\in (0, \infty) \end{aligned} \tag{1}$$

[1]https://github.com/librosa/librosa

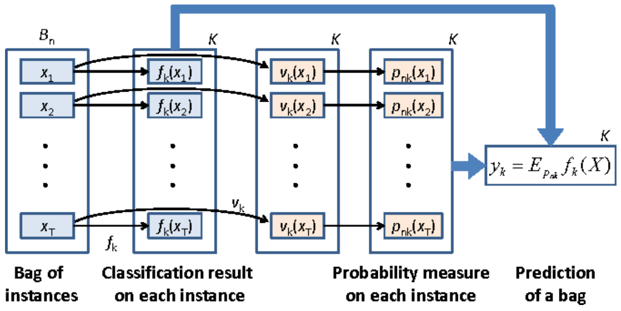


Figure 1: Attention model for Audio Set [4]

The (x_i, y_i) and (x_j, y_j) are two training and label pairs, which are randomly sampled form the training data. λ is drawn from a Beta distribution and lies in the region [0,1]. The experiments in the work [10] shows that increasing the α would increase the training error on real data and minimize the generalization gap. They also found that $\alpha \in [0.1, 0.4]$ would give an improved performance and large α will leads to underfitting.

3.4. Multi-level Attention Model

The dynamic changes are crucial in audio tagging challenges. Only considering the frequency structure might be useful for the recognition of the melodic instruments, but for the tags such as 'gunshort', and 'knock', the temporal changes are much more important. An attention model, which assign different weights for the instances in a time series, is a good strategy for considering the dynamics of sound. An attention model structure [4] presented for the AudioSet can be seen in Fig. 1. B_n represents an audio file with 10 instances $x_1 \sim x_{10}$, $f_k(x_n)$ represents the classification results for class K. The weights for each instance are firstly computed as $v_k(x_1) \sim v_k(x_{10})$ and then normalized to $p_{nk}(x_1) \sim p_{nk}(x_{10})$ so that $\sum_i^{10} p_{nk}(x_i) = 1$. Finally, multiplying the weights with the relevant prediction results and summing them together gives the final prediction for class K.

Combining the features from different levels and different timescale can provide more accurate descriptions. A CNN-based architecture [11] has shown great performance for music tagging by aggregating the multi-level and multi-scale features. Concatenated features extracted from different levels of CNN has also been proven useful in computer vision tasks [12]. Inspired by the works [4] [5] on Google Audio Set, we decided to choose a similar multilevel attention structure for audio. We did not try multi-scale methods because the fine-tuned VGGish models can only generate features for the 0.960 seconds fixed length audio. We prepared each training sample as 6 concatenated segments, each segment contains 0.960 seconds audio. Then the 6×128 dimensional features extracted using the fine-tuned VGGish model are fed into the neural network for training.

Fig. 2 shows the model structure. Each of the 6 instances are fed into a fully connected neural networks with 3 layers. Different color boxes represent the different levels features, and the features from the same level are then fed into an attention model, shown in Fig. 1, to get the predictions. We then concatenated all three predictions from different levels and forward them into the final classification layer with the softmax activation to get the 41 probabilities for the 41 classes.

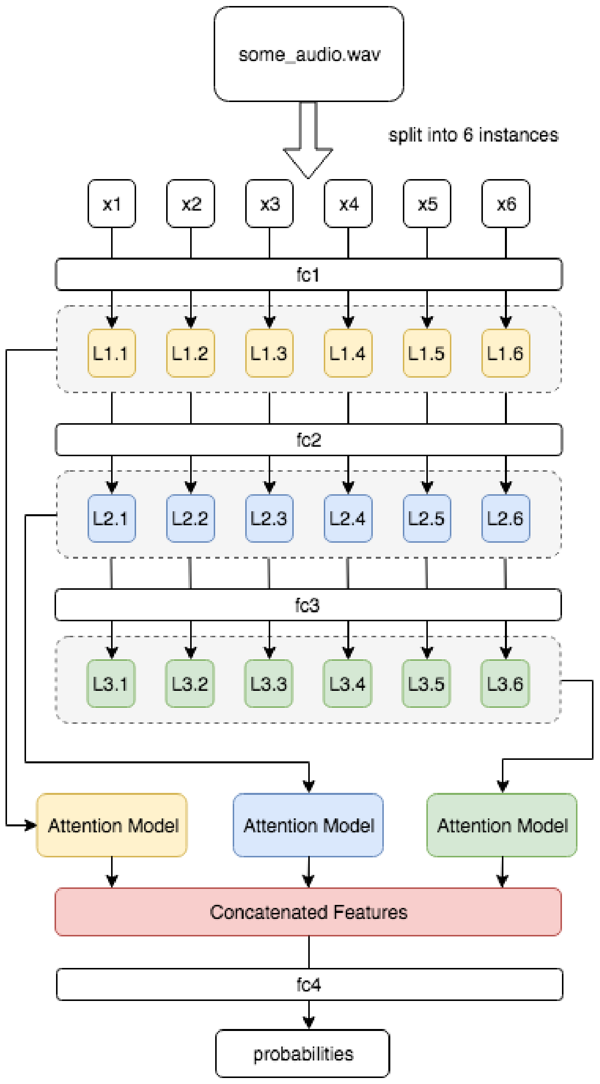


Figure 2: Model Structure for Multi-level Attention model

3.5. Evaluation Metric

The evaluation uses the Mean Average Precision @ 3 (MAP@3)[2]. The detailed implementation and explanation can be seen in the link provided in the footnote. Simply speaking, up to three predictions can be given even though there is only one correct label. The order of the predictions matters in this setting. If the correct label is predicted in the 1st position among those three predictions, the system would get a score 1. 2nd position would get a score 1/2, 3rd position would get a score 1/3. If there is no correct answer in the three predictions, the score would be zero. The average of the scores for all the testing samples would be the final evaluation score. However, the public leaderboard only has the score for around 19% of the testing data. So the scores in this paper are all evaluated upon

[2]`https://github.com/benhamner/Metrics/blob/master/Python/ml_metrics/average_precision.py`

Table 2: The one second DNN evaluation results on different fine-tuned VGGish models

Feature Extractor	MAP@3
Baseline	0.704
Original VGGish	0.606
Balanced First Part	0.870
Balanced Last Part	0.864
Balanced All	0.891
Unbalanced First Part	0.858
Unbalanced Last Part	0.872
Unbalanced All	0.892
Ensemble All	**0.903**

Table 3: Evaluation results on different model structures and hyper-parameters.

Index	Model Structure	MAP@3
A	1-Segment Multi-level	0.903
B	LSTM	0.914
C	6-Segment Multi-level Attention	0.925
D	C + Pitch shifting Augmentation	0.930
E	D + Mixup ($\alpha = 0.1$)	0.930
F	D + Mixup ($\alpha = 0.2$)	**0.936**
G	D + Mixup ($\alpha = 0.4$)	0.931
H	D + Mixup ($\alpha = 1.0$)	0.930

those 19% and the final results might be different.

4. EXPERIMENTS

4.1. Different fine-tuned VGGish models

Firstly, we tested the performance of different fine-tuned VGGish models on a simpler model structure. In this experiment, we built a simple 3-layer fully connected neural networks with each layer containing 600 units. The classification results from each layer are concatenated together as input to the final layer. The activation function is 'relu' for all the layers, except that the classification layers use 'softmax' activation function. Batch normalization and dropout with ratio 0.4 are used after each middle layer. The input is the 128 dimensional features for 0.960 seconds. For evaluation on files with different length, we took the average results as the score. As can be seen from Table 2, all of the fine-tuned models outperform the original VGGish model. 'Balanced first part' means the feature extractor is trained using mini-batch balancing on the first 8000 training samples and validated on the remaining ones. 'Balanced last part' means the feature extractor is trained using mini-batch balancing on the last 8000 training samples and validated on the remaining ones. 'Balanced all' means taking the geometric means of the results from the 'Balanced first part' and 'Balanced last part'. 'Unbalanced' means the model is trained without mini-batch balancing. The remaining names can be interpreted in similar way. There is no clear difference between the models trained with mini-batch balancing and those without. However, using geometric mean results of all 4 models will give the best results.

Table 4: Evaluation results on different number of segments multi-level attention models

Number of Segments	1	2	4	6
MAP@3	0.917	0.919	0.931	0.936
Number of Segments	8	10	Ensemble All	
MAP@3	0.929	0.931	0.931	

4.2. Different neural network structures and hyper parameters

Section 4.1 has shown that utilizing the results from all 4 fine-tuned models would give the best results. Thus for each model structure in this section, we also used the same strategy. The results for different model structures and different random mix factors α can be seen in Table 3. Model A has the same setting as the best results from Table 2. Model B uses one LSTM layer with 600 hidden units upon the original variable length input. Model C has the same 6-segment multi-level attention structures shown in Section 3.4, other parameter are same as A. Model D is based on model C with pitch shifting augmentation. Model E, F, G, H are based on D with different mixup factors.

From the comparison between model A, B, C, LSTM is better than the 1-segment multi-level model. The 6-segment multi-level attention model performs the best. Thus temporal information plays an important role in recognition, and multi-level attention model has better ability to model the temporal information than LSTM in our settings. Using the pitch shifting augmentation to generate a relevantly more balanced training data also improves the results a little bit. The comparison between model E, F, G, H shows mixup with the $\alpha = 0.2$ has the best performance among these 4 choices. However, since the difference among C-H is quite small, further significance test might still be needed.

4.3. Different number of segments

We also tried the multi-level attention model with different number of segments as input. Other training settings, such as the pitch shifting augmentation and mixup, are the same as the best results in Table 3. Results can be seen in Table 4. For the ensemble results, we took the geometric mean of all the results. We can see that the 6-segment model has the best performance and the ensemble does not improve the overall score. The reason might be that these models are not diverse enough.

5. CONCLUSIONS

In this paper, we tried different fine-tuning methods on the AudioSet VGGish model for generating 128 features for 0.960s audio. The results show that the combination of the 4 models trained with different train-validation splitting and balanced/unbalanced techniques would give the best results. We also implemented different neural network structures for comparison and found that multi-level attention model performs the best among all. This shows the importance of modeling the temporal information. The 6-segment multi-level attention model with pitch shifting augmentation and mixup method using $\alpha = 0.2$ has the best MAP@3 at 0.936 in public leaderboard. Further research might include more thorough fine-tuning, building own CNN model for feature generation, utilizing multi-scale features along with multi-level features for the attention model.

6. REFERENCES

[1] H. Aronowitz, “Segmental modeling for audio segmentation,” in *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 4. IEEE, 2007, pp. IV–393.

[2] A. Mesaros, T. Heittola, A. Diment, B. Elizalde, A. Shah, E. Vincent, B. Raj, and T. Virtanen, “Dcase 2017 challenge setup: Tasks, datasets and baseline system,” in *DCASE 2017-Workshop on Detection and Classification of Acoustic Scenes and Events*, 2017.

[3] J. F. Gemmeke, D. P. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, “Audio set: An ontology and human-labeled dataset for audio events,” in *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*. IEEE, 2017, pp. 776–780.

[4] Q. Kong, Y. Xu, W. Wang, and M. D. Plumbley, “Audio set classification with attention model: A probabilistic perspective,” *arXiv preprint arXiv:1711.00927*, 2017.

[5] C. Yu, K. S. Barsim, Q. Kong, and B. Yang, “Multi-level attention model for weakly supervised audio classification,” *arXiv preprint arXiv:1803.02353*, 2018.

[6] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.

[7] S. Hershey, S. Chaudhuri, D. P. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, *et al.*, “Cnn architectures for large-scale audio classification,” in *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*. IEEE, 2017, pp. 131–135.

[8] E. Fonseca, J. Pons, X. Favory, F. Font, D. Bogdanov, A. Ferraro, S. Oramas, A. Porter, and X. Serra, “Freesound datasets: a platform for the creation of open audio datasets,” in *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR 2017)*, Suzhou, China, 2017, pp. 486–493.

[9] E. Fonseca, M. Plakal, F. Font, D. P. Ellis, X. Favory, J. Pons, and X. Serra, “General-purpose tagging of freesound audio with audioset labels: Task description, dataset, and baseline,” *arXiv preprint arXiv:1807.09902*, 2018.

[10] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, “mixup: Beyond empirical risk minimization,” *arXiv preprint arXiv:1710.09412*, 2017.

[11] J. Lee and J. Nam, “Multi-level and multi-scale feature aggregation using pretrained convolutional neural networks for music auto-tagging,” *IEEE signal processing letters*, vol. 24, no. 8, pp. 1208–1212, 2017.

[12] X. Meng, B. Leng, and G. Song, “A multi-level weighted representation for person re-identification,” in *International Conference on Artificial Neural Networks*. Springer, 2017, pp. 80–88.