Enhancing and Re-Purposing TV Content for Trans-Vector Engagement (ReTV)
H2020 Research and Innovation Action - Grant Agreement No. 780656

Enhancing and Re-Purposing TV Content for Trans-Vector Engagement

Deliverable 1.3 (M30)

Data Analysis and Annotation

Final Version

This document was produced in the context of the ReTV project supported by the European Commission under the H2020-ICT-2016-2017 Information & Communication Technologies Call Grant Agreement No 780656

DOCUMENT INFORMATION

Delivery Type	Report
Deliverable Number	1.3
Deliverable Title	Data Ingestion, Analysis and Annotation
Due Date	M30
Submission Date	June 30, 2020
Work Package	WP1
Partners	CERTH, MODUL Technology, webLyzard
Author(s)	Konstantinos Apostolidis, Nikolaos Gkalelis, Evlampios Apostolidis, Vasileios Mezaris (CERTH), Lyndon Nixon, Adrian M.P. Braşoveanu, Jakob Steixner (MODUL Technology), Katinka Boehm (webLyzard)
Reviewer(s)	Arno Scharl (webLyzard)
Keywords	Data Ingestion, TV Program Annotation, TV Program Analysis, Social Media Retrieval, Web Retrieval, Video Analysis, Concept Detection, Brand Detection
Dissemination Level	PU
Project Coordinator	MODUL Technology GmbH Am Kahlenberg 1, 1190 Vienna, Austria
Contact Details	Coordinator: Dr Lyndon Nixon (nixon@modultech.eu) R&D Manager: Prof Dr Arno Scharl (scharl@weblyzard.com) Innovation Manager: Bea Knecht (bea@zattoo.com)

Revisions

Version	Date	Author	Changes
0.1	19/5/20	V. Mezaris, K. Apostolidis	Created template and ToC
0.2	9/6/20	L. Nixon	Added MODUL sections
0.3	11/6/20	N. Gkalelis	Added sections 4.2.1, 4.2.2 and 4.2.3
0.4	12/6/20	K. Apostolidis, E. Apostolidis	Added sections 5, 4.1 and 4.2.1
0.5	15/6/20	E. Apostolidis	Added Introduction and Conclusions text
0.6	16/6/20	V. Mezaris	Reviewed the whole text
0.7	17/6/20	L. Nixon, A. Brasoveanu	Re-organized MODUL sections
0.8	19/6/20	A. Scharl	QA review
0.9	23/6/20	E. Apostolidis, L. Nixon	Post QA updates
1.0	26/6/20	L. Nixon	Final check

Statement of Originality

This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both.

This deliverable reflects only the authors' views and the European Union is not liable for any use that might be made of information contained therein.

Contents

EXECUTIVE SUMMARY

This deliverable is an update of D1.2. It covers the topics of annotation and knowledge graph alignment, concept-based video abstractions, and brand detection. Additionally, despite the content collection across vectors task not being active in months M21 to M30, it reports the work done on adding scenario-specific terms, accounts, Websites and the classification of sources to allow for orthogonal organization. Concerning annotation and knowledge graph alignment, it updates on the implementation of an accurate NLP & NEL pipeline for annotating the collected data with respect to keywords and Named Entities and aligning annotations to our Semantic Knowledge Base (SKB) as well as external knowledge sources. With respect to concept-based video abstractions, it presents the results of our efforts on optimizing the speed of the ReTV Video Analysis service, the developed deep neural network pruning techniques for improving the performance of concept detection, and the extension of the set of concept pools that are supported by the ReTV Video Analysis service. Concerning brand detection, this deliverable discusses our performance optimization efforts, the extension of the set of recognizable brands, and a method for car brand detection. Finally, this document also updates on the ReTV WP1 Video Analysis REST service usage instructions, API and reports new functionality.

ABBREVIATIONS LIST

Abbreviation	Description
API	Application Programming Interface: a set of functions and procedures that allow the creation of applications which access the features or data of an application or other service.
DNN	Deep Neural Network: an artificial neural network .
DCNN	Deep Convolutional Neural Network: a type of artificial neural network.
HTTP POST/GET	Types of method in the Hypertext Transfer Protocol (HTTP). The HTTP POST method is used to send data to a server to create/update a resource. The HTTP GET method is used to request data from a specified resource.
IPTV	Internet Protocol Television: is the delivery of television content over Internet Protocol (IP) networks.
JSON	JavaScript Object Notation: a data-interchange format.
LSTM	Long Short Term Memory networks: a type of recurrent neural network.
MTL	Multi-task learning: a field of machine learning in which multiple learning tasks are solved at the same time, exploiting commonalities and differences across tasks.
NEL	Named Entity Linking
NER	Named Entity Recognition
NLP	Natural Language Processing: subfield of linguistics, computer science, and artificial intelligence concerned with the interactions between computers and human (natural) languages.
OTT	Over The Top: content providers that distribute streaming media as a standalone product directly to viewers over the Internet, bypassing telecommunications that traditionally act as a distributor of such content.
RDF	Resource Description Framework: a method for conceptual description or modeling of information that is implemented in Web resources.
REST	Representational State Transfer: an architectural style that defines a set of constraints to be used for creating Web services.
RNN	Recurrent Neural Network: a type of an artificial neural network.
SKB	Semantic Knowledge Base: a RDF-based triple store for a knowledge representation of keywords and entities during in annotation of documents
TVoD	Transactional Video on Demand: a distribution method by which customers pay for each individual piece of video on demand content.
URL	Uniform Resource Locator: a reference to a Web resource that specifies its location on a computer network and a mechanism for retrieving it.

1. Introduction

This deliverable reports on the work done from M21 to M30. In this reporting period the following task were active: Concept-Based Video Abstractions (T1.2), Brand Detection in Video (T1.3), and Annotation and Knowledge Graph Alignment (T1.4). The task of Content Collection Across Vectors (T1.1) had ended on M20, however we continued to monitor and maintain the data collection pipeline. Section 2 reports the collection results. Regarding the Annotation and Knowledge Graph Alignment, Section 3 reports the various techniques introduced to improve keyword relevance, including alignment of extracted keywords, and the combination of keyword detection and Named Entity Recognition pipelines. Section 4 reports the work done regarding video fragmentation and concept-based annotation, which includes implementation-related optimization efforts, as well as algorithmic optimization, specifically, the implementation of pruning techniques for deep neural networks. Additionally, various new functionalities that were implemented to support specific ReTV use-case scenarios are described in detail. We report all updates in the direction of brand detection in Section 5, where, again, the optimization of performance was the focus. Furthermore, we describe the introduction of a new car brand detection sub-module in the Video Analysis services, after discussions with partners. The deliverable concludes (Section 7) with a brief summary and concluding remarks.

2. Content Collection Across Vectors

2.1. Status

Our data collection pipeline continues to be available in 4 languages, across 3 social media platforms and crawling several hundred Websites as well as daily EPG data since September 2018. In one week, we collect on average 250 000 news articles, 4 000 Webpages about TV/radio brands, 300 000 social media posts about TV/radio brands and 2 000 EPG items. ReTV clients now have access to over 22 million news documents, 280 000 TV/radio specific Web documents, 16 million TV/radio specific social media postings and 80 000 EPG items that can be browsed and analysed in the TVP Visual Dashboard. The data collection is regularly quality checked and drives our data annotation and analytics, contributing to the prediction and recommendation services.

2.2. Updates from Previous Report

Compared to the source statistics reported in Deliverable D2.2, we had added collection of Dutch Web and social media content based on selected TV channels and programs (accounts and terms monitoring). Since initiation of Dutch content collection in July 2019, we can report in 11 months (to 9 June 2020) that we collected 734 000 news articles, 264 000 tweets, 48 000 TV/radio Webpages, 31 000 Facebook posts and 1 400 YouTube videos. In all language data sources, we have added further Websites, social media accounts or terms for monitoring as required by our use-case partners or new scenarios arising via external stakeholders. For example, early in 2020 we started to monitor accounts and terms related to the Eurovision Song Contest 2020 with the expectation to use the resulting data analysis in May. Unfortunately the event was cancelled, and also since March 2020 we started monitoring terms related to the corona virus / COVID-19 as it was in any case dominating our news media sources.

2.2.1. Source Classification

In expanding further the data sources we collect for various scenarios in ReTV, we realised that the core separation of the sources into three buckets: (News) Media, TV/Radio and Misc (other) was too simplistic for configuring the dashboards for different stakeholders (this is a different separation than the implementation of the so-called 'mirrors' we use for each data source, which fundamentally splits into (1) a crawler for Web sites, (2) API-specific code for each social media platform and (3) a dedicated API to receive statistical or metadata documents pushed from an external source such as the EPG data and audience metrics from GENISTAT). We decided it would be beneficial to provide an additional classification of each source that could be orthogonal to the source buckets as a future-proof strategy as the data source collections grow further and in different directions for each scenario. Then, in a particular scenario configuration, it should be simpler to switch a view on data to all sources relevant to a particular classification regardless of whether it is collected from the Web, social media or other and whether it fits into the main split of (News) Media, TV/Radio or Misc (other).

2.2.2. Scenario Data Collection: Fritz, EUScreen and Europeana

We collected new data sources for selected partner scenarios: Fritz (youth radio station which is part of RBB), EUScreen (a portal that offers free online access to European AV archives) and 'Summer of Sport' (a new editorial campaign by Europeana). We initiated a process for externals to provide data sources. We created a data source collection sheet where stakeholders

can identify the Websites, social media accounts and terms we should collect from. This sheet covers for Websites: Website title, domain address (base URL), language, country, data source (Media, TV/Radio, Misc), classification (free text), URL pattern for Web crawl. The URL pattern is necessary for the correct Web crawling, e.g. the Website of VICE magazine (`https://www.vice.com`) places all of its news articles under the sub-folder `https://www.vice.com/de/article/*` where * is the wildcard (match all). This ensures that only the articles would be crawled and not other Website sections like a magazine subscription etc. For social media, there is both a term list (which can use regular expressions and is applied for all social media platforms whose APIs allow search over public content such as Twitter and YouTube) and accounts list per platform. In the sheet, for each account the following information must be given: URL of the account, language, country, data source and classification. In Facebook, the 'account' must be a publicly accessible Facebook Page - we can not collect from individual's Facebook accounts nor from Facebook groups.

While classifications are free text in the sheet, the intention is that consistent labels are used - for Fritz, we marked sources particularly relevant to its young audience as 'youth'. For EUScreen, we marked sources particularly focused on cultural heritage content as 'culture'. For Europeana, we found many sources useful for the sports content were already being collected (contemporary news sources cover sports) but added sports-specific sites and accounts with the label of 'sport'. As such, it is helpful to suggest to the external organizations in advance which labels to use for their types of content. We double checked all provided lists as the 'data experts' before adding them to the ReTV data ingestion pipeline, e.g. if a Website is publishing relevant items or a social media account is active.

As a result, for Fritz we added 41 Websites, 30 Facebook pages, 20 Twitter accounts and 31 YouTube channels, the majority of which classified as 'youth'. It should be noted that externals tended to also include in their lists some news and TV content sources that we were already collecting. For EUScreen, we added 33 Websites, 22 Facebook pages and 76 Twitter accounts, again the majority classified as 'culture'. We were still waiting on the lists from Europeana at the time of writing.

2.3. Evaluation of Data Quality

Whenever new data sources are added, we take care to allow a few days for ingesting first documents from the source then check manually the quality, i.e. if there is a correct title and description text, that the description text is sufficient and relevant, that we can see keyword and entity extraction being done on the text.

We particularly mark any sources which are empty after several days for a specific check as this can either mean the source is not active (if it is completely dormant, we remove it from the collection pipeline) or the collection from the source is broken.

Checking with the newly added sources (74 Websites, 179 social media accounts) we found that the experiences learnt from the initial TV/Radio data collection quality checks and corrections had paid off in that our already configured mirrors worked very well 'out of the box' to collect data correctly from the new sources. Only one Website did not work with the crawler; two were removed as we found they were not posting any new content. Nine Websites were corrected in terms of the right URL paths to collect data from. Regarding social media, the correctness of the data is related solely with the API call to the platform, so as long as the API does not change and the correct identifier is used for the chosen account, we find that the data collection runs smoothly for all accounts, regardless whether it is Twitter, YouTube or Facebook.

3. Annotation and Knowledge Graph Alignment

3.1. Status

After documents are collected, they are annotated according to our evolving knowledge graph (the Semantic Knowledge Base, or SKB) for Named Entities and keywords (significant words or terms which are not named entities). This step greatly assists the subsequent analyses of the data at scale, such as uncovering trending topics (which can be seen as a cluster of entities and/or keywords). In this period, we have significantly involved both pipelines. In keyword extraction, various techniques were introduced to strongly improve keyword relevance, including alignment of extracted keywords (n-grams) with the labels of named entities in our SKB. For this, the keyword detection and Named Entity Recognition (NER) pipelines were combined, where also our work on NER/NEL has progressed in accuracy in part through the support of the ORBIS visual evaluation system. Since this merged approach relies on a comprehensive and error-free SKB, processes were introduced to both internally and externally clean the metadata collected and replicated in the SKB with a Web interface made available to ReTV partners to either query and browse entities in the SKB (read-only SKBBrowser) or also allow expert users to edit and save the metadata for entities in the SKB (read/write SKBEditor).

3.2. Updates from Previous Report

3.2.1. Natural Language Processing (NLP)

We have launched a new pipeline for keyword detection which is now combined with the named entity recognition to allow for a further disambiguation of keywords (which are arbitrary n-grams) through the labels of the named entities in the SKB. The intention is that prior cases of having multiple partial matches to named entities, something which typically happens with names of people, organisations and locations which are made up of several words together, is replaced by a single keyword which is aligned to that single Named Entity (e.g. 'donald', 'president donald', 'president trump' should all be aligned to Donald Trump).

The next issue we have been addressing with keywords is the translation between languages. After all, a user may explore keyword-based associations in their own language regardless of the original language of the source document, which requires that all keywords are mapped to a single language to be aggregated. The above-mentioned alignment to labels of Named Entities already helps this process since some labels are language independent (like 'Donald Trump', avoiding various translations of 'president' that might appear in the original n-grams) and others are mapped together to a single Named Entity in the SKB allowing for an equivalence to be made (e.g. place names which vary according to language). Entity labels in different languages are generally sourced from the external knowledge graph used to feed the SKB, e.g. WikiData, whereas we now have the additional capability to curate these labels by ourselves using the SKBEditor tool (see later in this section).

Regarding the more general keywords (n-grams) which are not aligned to entity labels, we store our keywords also in the SKB and connect them to additional grammatical, word sense and translation information. The grammatical and word sense information for matching keywords comes from OmegaWiki, automatically curated further by our own NLP processes (e.g. lemmatization and stemming the surface form, comparing with the text in our own document repositories for other terms with a small word distance - this also uncovers small spelling differences like 'honour' and 'honor' as well as common misspellings). We generate machine translations of keywords by calling the Google Translate API the first time a new keyword is

detected through the pipeline for the missing target languages (out of EN, DE, FR and NL with the source language naturally removed) and caching the translation results in the SKB.

This approach is not without its errors, especially when keywords are ambiguous to the translation process and therefore, as with all our data processes, we have a policy of sampling results (which we can do through the TVP Visual Dashboard) to identify any issues and debug them. Where necessary, we manually correct translations inside the SKB.

In the result, keywords may still occur which are 'correct' in the sense that they represent a re-occurring n-gram in the textual corpus but are irrelevant because they do not represent a meaningful association for the data analysis. An example coming from the media sources would be 'test subscription' which is repeated across many Web pages for media organizations and thus becomes a keyword but is not useful in the data analysis. In a first stage, we identified such keywords within the separate data sources set up for use-case partners RBB and NISV respectively in their ReTV dashboards and created individual stoplists of keywords that should be removed from the keyword detection results.

We are now looking at the creation of a global keyword stoplist that can be applied to all results of our keyword detection pipeline, taking care to not include any keyword that could prove useful for any data analysis task.

3.2.2. Semantic Knowledge Base (SKB)

The Semantic Knowledge Base (SKB) has evolved in the past year, with a major difference being our choice to use WikiData as our 'base' knowledge graph for sourcing Named Entities instead of DBpedia. We had started when we added Events to the SKB (see ReTV deliverable 2.1, Section 2.3). We preferred WikiData as an external events source as we found the data available to be better structured and cleaner than the equivalent in DBPedia. Following that, we began to also align our internal sets of Person, Organization and Location entities to the entity metadata in WikiData (Location entities are first linked to Geonames entities as it has the best coverage of geo-locations. Following that, 'sameAs' relations are created to the equivalent WikiData entity when it exists). The expected advantages include that WikiData has a better coverage of entity labels, both within and across languages, as well as that the entity metadata available via the WikiData API is much more up to date (DBPedia provides via its API the data from the most recent dump from Wikipedia which might be 6 or more months out of date, although we acknowledge there is also an additional live.dbpedia.org endpoint which should return on-the-fly knowledge extracted via the current Wikipedia content).

One of the functionalities of WikiData which we began to make use of in the event extraction work is that it provides a Recent Changes API which shows which entities have changed (we focus on the last 30 days and check every 30 days). We use this to keep the event entity information in the SKB more up-to-date.

Another update to the SKB was to include a thumbnail property for entities pointing to a Wikimedia URL where a thumbnail image for the entity could be found. This was found using WikiData properties like wdt:P18 (thumbnail), wdt:P154 (logo) or wdt:P41 (flag image). The thumbnail can be used in the display of entity information, and it has been integrated into the entities view on the TVP Visual Dashboard.

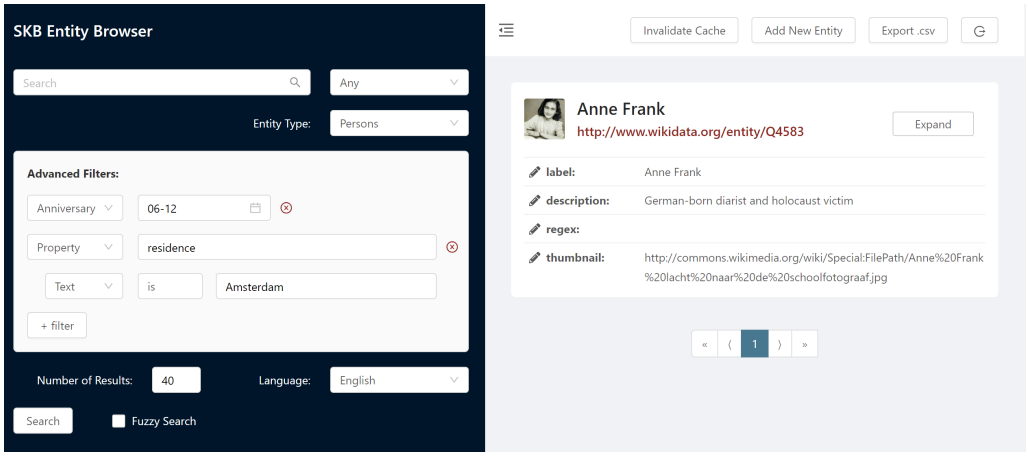

Figure 1: Screenshot of the updated SKBBrowser

3.2.3. SKBBrowser

In deliverable D2.2, we presented the first version of the SKBBrowser. One year later, we can report on significant improvements in both the user interface and in the search and result display functionalities. Figure 1 shows the new interface and some of the functional extensions are also visible.

In the browser, text search can be made across all SKB entities, either matching the search terms to any property value (of type 'string'), prioritizing exact value matches and multiple matches for an entity, or in a specific property value (name, description, URI). It can also match on all entity types, or on a specific entity type (Persons, Organizations, Locations, Events as well as the NonEntityKeywords). Fuzzy search may be selected to match on strings which have a close word distance from the search string - we use the Levenshtein distance which allows for one edit for word length 3-5 characters and two edits for words of length $>$ 5. Results are ranked according to exact (phrase) matches at the top, then prefix matches (the search term prefixes a longer string), then more general matches (the search term occurs within a longer string).

Search can now also be managed using the search filters which provide for additional restrictions on matching results. There are three filters available in the SKBBrowser: Date Range, Anniversary and Property. The date range filter allows the specification of absolute start and end dates (year-month-day) and matches entities which have a date property whose value occurs within that date range (i.e. start and end dates of events, birth/death date of persons, inception date of companies). The anniversary filter allows the specification of a calendar date (month-day) without year and returns entities which have an anniversary on that date - this is likewise for Persons matching on birth and death dates and for Organizations the inception date (the founding of the company) and for Events if it is within the date range in which the event occurred (past values only, as we also have future events in the SKB). The property filter allows for a match on a specifically named entity property and its given value. Supported values for matching are string (text), date and number, so the named property must have one of those types. Text matching can include ('is') or exclude ('is NOT') matches in results. Date matching can be to match a given date exactly, or all dates up to and including that date, or all dates from that date onwards. Number matching can also be a match on the exact value, all values up to and including the value, or all values from that value onwards.

Search results have also been improved. Logic was added to better handle events which occur over longer time frames, including them in search results but ranking them lower in results lists than events with more specific dates. Wikidata lookup was used to include human understandable labels for the property values which use Wikidata URIs, e.g. the place of birth is now shown as 'Honolulu' and not just 'wd:Q18094'. Finally, we looked at various heuristics for the ranking of the results: text search results rank entities higher which better match the text search term; with a date range, results are ordered by matched date. For multiple search criteria, entities which match more criteria are ranked higher. Specifically for the 'all' option, matches on name fields (preferred name, rdfs:label, skos:altLabel etc.) are scored higher while matches in URIs (image, website) are scored lower for ranking.

Other improvements made to the SKBBrowser were that a CSV export function was added for the search results and support for user roles included, so that different users could access different functionalities (this has been used to allow for expert users with SKB write permissions).

Our new SKBEditor extends the SKBBrowser to allow users with the correct rights to also add entities or edit the entity metadata. There is a new 'Add New Entity' button available to them where an entity of type Person, Organisation or Location may be added by providing a set of property-value(s) pairs - the entity will be allocated its own URI within the SKB's namespace `http://weblyzard.com/skb/`. Also entities displayed in the search results have edit options alongside some of their properties (e.g. for Persons: label, description, regex, thumbnail, preferred name) - we focus on properties with textual values as they are simpler to edit directly within the Web interface and are more likely to be useful in being edited as they are the properties currently displayed in the entity view of the ReTV dashboard (e.g. to expand a description of an entity). The exception is the thumbnail property, whose value is an URI, however again there will be entities without thumbnails available via Wikimedia or where a better thumbnail might be available, therefore we allow editing of this property too. This necessitated changes in the implementation of the SKB as we added the ability to manage provenance information for the manual edits, i.e. capturing that a triple in the RDF store (a subject-property-value statement) was added by a certain user at a certain time, while not deleting previous triples including the information sourced directly from the external knowledge graph such as WikiData. This allows us also to rollback changes in the case that the manual edit proves to be incorrect or just to allow different views on the data - e.g. one user may want to explore the entities based purely on the WikiData information, another may prefer to see the manual edits by a certain expert user when present.

We have also implemented a means to import entities into the SKB via CSV files but this is currently restricted to a manual import process with supervision since it represents a larger change to SKB content in one step; it also relies on the correct preparation of the CSV to avoid errors in the resulting SKB update. We used this batch import for example to add 281 'world days' and 79 other 'awareness days' to the SKB.

3.2.4. Recognyze NER/NEL

In the past year, we have released an updated version of our graph-based NER/NEL engine known as Recognyze, which we have internally termed **Recognyze-NG** (for Next Generation). Recognyze uses graph-disambiguation techniques, in particular based on community detection algorithms (e.g., Louvain clustering algorithm) and ML learning (e.g. name variants generation using SVMs), in order to correctly identify, classify and link entities to their targets from various Knowledge Graphs.

Several low-level components can be used when defining the lexicons and profiles that will be used for extracting the entities from the text:

- **Linked Data Sources** like Wikidata or DBpedia are aligned with the entity informatin in the SKB.
- **Filters** can be used for removing bad URIs (e.g. broken).
- **Preprocessors** are used for specific tasks like extracting abbreviations (e.g., stock tickers), removing noise or limiting the minimum character count.
- **Analyzers** receive a set of name variants and return only those that match the search criteria (e.g., entities with certain types).

A set of higher-level modules use the profiles built with the previously mentioned components in order to perform a set of Information Extraction tasks:

- **Configuration Component** defines the low-level search configuration via a set of JSON files.
- **Candidate Searcher** scores the best candidates for a certain query.
- **Disambiguation Component** runs a set of algorithms and delivers the annotations.

One of the aspects of Recognyze-NG is that we have moved from using DBPedia or Wikidata as our external targets for Named Entity Linking (NEL) process to using identifiers from our own SKB Knowledge Graph which is aligned to WikiData as an external KG source. This step was implemented as it both (i) helps us reduce the dependence on multiple knowledge graphs; (ii) allows us to fine-tune the graph as we want for our internal tasks (e.g., in case of Recognyze by directly providing the added naming variants from a single source instead of merging them from DBpedia, Wikidata or Geonames which made the debugging of the respective profiles rather tedious).

Recognyze was developed in an iterative manner. Most of the insights we needed in order to develop it were obtained while inspecting Recognyze run results with **Orbis**, a visual benchmarking tool that we have developed for comparing results with gold standards and with similar NEL engines. The initial version of Orbis included the possibility to visually inspect the results of the evaluation, as well as the possibility to compare the results of different NEL engines.

The main Orbis interface is document-centric and displays both the gold standard and the annotator result (see Fig. 2. Two viewing modes are available: *standard* and *dark* mode.

In order to enable users to perform a wide range of analyses, a button group allows selecting the desired entity classification scheme. These schemes are paired with coloring schemes which support the users in quickly identifying problematic results. Currently, Orbis supports the following classification schemes:

- *Entity* - each distinct entity is presented with a different color. When examined across both panels (*gold* and *predicted*) the entities displayed with matching colors were correctly identified.
- *Type* - entities are classified by type. The coloring represents the typing in both panels.
- *Result* - the classification and coloring scheme reflects the test results (e.g., TP, FP, TN, FN). This coloring mostly affects the *predicted* panel.

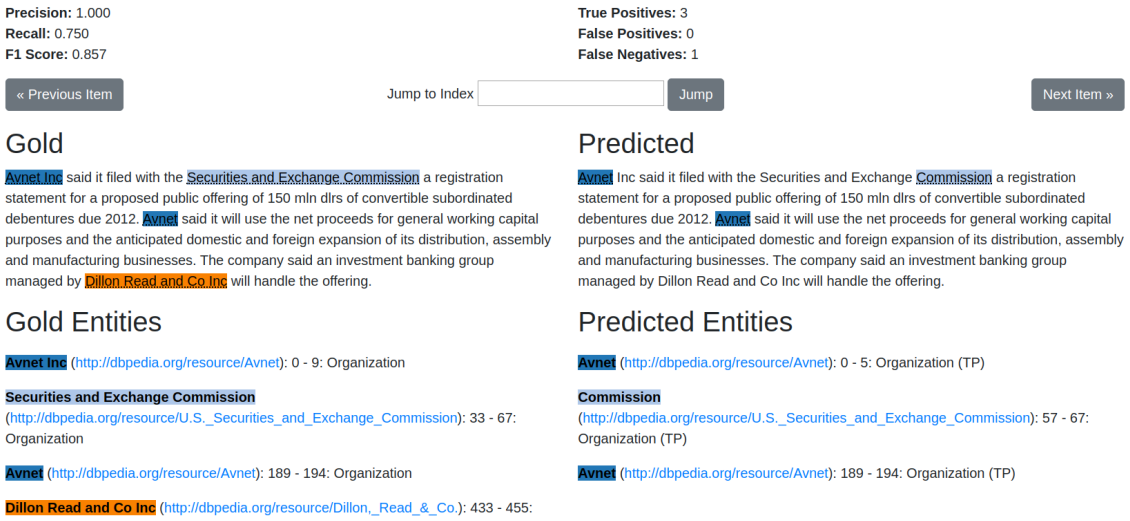

Figure 2: Recognyze results for document 41 from Reuters corpora with overlap settings.

Advanced classification schemes have also been developed for various plugins, but they have more dependencies (e.g., evaluation types, taxonomies, corpora) and are enabled only for special evaluation types (e.g., slot filling). Examples of such classifications include:

- *Cluster* - is a special classification mode that can be used only for slot filling evaluations. All the attributes that belong to the same entity are included in the same cluster. For example, if a text contains information about a *company*, its *CEO* and its *address* - they will all be included in the same company cluster and colored with the same color.
- *Error* - is a special classification that displays classification errors according to a taxonomy or ontology. The default taxonomy for Orbis is currently based on [3]. Each error class is assigned a different color.

In addition to the *view* pages, Orbis also provides an *overview* which offers additional information related to an evaluation. Besides the general results, this view describes the evaluation settings (e.g., evaluation type, tool(s), datasets, etc) and offers some *top k-lists* (e.g., top performing documents, worst performing documents) that help developers in quickly navigating results. For debugging reasons, a reduced set of this functionality (e.g., only general settings and results) can be displayed on each page.

Together with the improved interface comes the possibility to perform several different evaluation types:

- *Named Entity Recognition (NER)* - takes into account entities and their types only;
- *Named Entity Linking (NEL)* - is the classic Orbis evaluation which also takes into account the linking process (e.g., if entities from the gold and those returned by the annotators have the same links);
- *Slot Filling (SF)* - it is essentially an evaluation focused on relation extraction. This type of evaluation is fundamental for all the services built on top of Recognyze (e.g., extraction of Web addresses or company information from Websites).

It has to be noted that Orbis was initially designed for performing only NEL evaluations, therefore the main criteria for evaluations is generally the link correctness. Since we are

continuously expanding the number of available evaluations, we have also provided different evaluation settings (e.g., strict, overlap, combined, etc), but as far as NEL evaluation goes, the links still need to be correct for a result to be a evaluated as a True Positive (TP), as it can easily be seen in Figure 2 (e.g., the link for *SEC* is correct in both gold and predicted results from the system, even if the system does not return the full surface form).

We continuously add new evaluation services to Orbis. Our goal is to eventually reach a point where we can evaluate most of our essential annotation services from lower-level ones (e.g., content extraction) to upper-level ones (e.g., relation extraction, sentiment analysis / affective classification) using this tool and provide both quantitative (e.g., metrics, statistics about features) and qualitative insights (e.g., through visualizations, details on the various error classes encountered) to help reach our goals. The next category of evaluations we plan to tackle is the *Content Extraction and Classification (CEC)*. It is a larger class of evaluations focused on the text extraction and classification pipelines which includes forum extraction (e.g., extraction of forum posts or of social media posts).

3.3. Evaluation of Keyword and Annotation Quality

3.3.1. Keyword Evaluation

We invited our use case partners RBB and NISV to perform a sample-based evaluation of the quality of the extracted keywords via the ReTV visual dashboard, following the updates to our keyword detection pipeline. This was particularly useful to test the NLP in non-English languages. RBB evaluated keywords from both English and German texts translated into German. NISV evaluated the keyword extraction from Dutch texts which are translated for the dashboard interface into English.

RBB was asked to use the dashboard in the Prediction Mode (looking at keywords projected to be important in future dates) with the interface language German. Prediction data sources were in both English and German. To emulate the scenario of using Topics Compass for prediction of future topics, the future date ranges were suggested to be the next 7 days. The list of associations in the top left of the dashboard interface are inspected and a spreadsheet used to evaluate them (the associations list for any one search will contain up to 20 keywords). They were asked to note how many appear correct and which of those belong in a keyword stoplist (i.e. the word or words are not useful for the prediction task). For incorrect keywords, the type of keyword detection error is noted after reference to the surface form (the text in the original documents from which the keyword is detected). Errors are: wrong translation (from English into German), wrong keyword (based on the surface form, e.g. just a part of the right term) or keyword as entity (keyword instead of an entity, e.g. the surname of a person as keyword when it should be their full name as entity). RBB conducted 7 searches and evaluated from 124 inspected keywords 111 as correct (90%) although 14 belong to the stoplist in their opinion. Only one translation error was reported whereas the remaining 12 errors were all cases of keyword as entity. Here, while we could observe a few cases where the surface form did not map correctly to a known entity (the clearest example was 'fsv mainz' and 'mainz 05' keywords rather than '1. FSV Mainz 05' entity), we found the majority (7 from the 12) to be gaps in known entities in the SKB. For example, persons like Sorona Dodd or Karsten Amman were not found as they are not in the SKB. The latter, Mr Amman, is a case of particular note as there is no external (WikiData) entity for this local Green politician in Germany. Here, the ability to add entities ad hoc to the SKB for particular scenarios (such as local Berlin politicians for RBB) is important, as otherwise there can not be a correct (entity-based) identification of references to these otherwise unknown or non-existent entities.

In the case of NISV, Dutch content is only visible in the exploration mode (past dates) where the interface language is set to English. Only Dutch sources were selected so all keywords displayed in English in the dashboard interface would be translated from the Dutch (actually we did find some references to English words in the Dutch documents, e.g. they tended to use the English titles for well-known films and books). They were asked to explore the data through the selection of their cultural heritage-related bookmarks as this would be closest to the use of the Topics Compass by them and their stakeholders (Europeana, EUScreen) with arbitrary date ranges. They conducted 11 searches and evaluated a total of 196 keywords. Of this, 156 were considered correct (79%) although 12 should be part of a stoplist in their opinion. Of the 40 keywords marked as errors, only 10 were translation errors (which suggests that the Dutch to English translations are correct to a very large extent, around 95%). From the other errors, 12 were wrong keywords and 18 were keyword as entity errors. Wrong keywords were sourced in ambiguities in Dutch, for example Dutch 'meet' is translated as English 'measure' but we had one case where the reference used the English term 'meet up' (a get-together). In all other cases, that ambiguity was tied to entities just as all cases of keyword as entity error, suggesting - as with the RBB evaluation - that the vast majority of issues arise from gaps in the SKB when disambiguating keywords as entities. For example, one wrong keyword 'shipping museum' came from the surface form 'Scheepvartmuseum' - actually a correct translation but the name belongs to a specific museum organisation in Amsterdam whose official English name is the National Maritime Museum. Others came from persons names, where only the surname was detected as a keyword and then the form 'van X' was translated as 'from X'. Since we will not have all Dutch persons in the SKB when considering arbitrary news or TV Web documents, we need to consider how we might include heuristics in the Dutch NLP to deal with such cases of word ambiguity.

3.3.2. Recognyze General Evaluation

Several annotator tools have already been integrated using the NIF output provided by their public endpoints. Not all annotators publish their best settings necessarily, some of them advising users to experiment until they find the best settings for their experiments (e.g., AIDA).

Besides Recognyze, the following tools have been used during our general-purpose evaluation:

- **DBpedia Spotlight** [7] is a statistical NEL engine that was originally built as a demo for showcasing DBpedia capabilities and ported to multiple languages. The statistical models from Spotlight are really good for larger Knowledge Extraction or WSD challenges where all words need to be linked to their respective KG entities, but they are not necessarily fine-tuned for typed NEL challenges.
- **Babelfy** [34] draws upon graph-based disambiguation algorithms and was built specifically for WSD tasks, but also performs relatively well for Wikipedia or DBpedia based-NEL tasks.
- **AIDA** [20] uses graph-based disambiguation algorithms and was one of the best NEL engines focused around Wikipedia linking.
- **Recognyze** [54] is a multi-KG (e.g., DBpedia, Wikidata, Wikipedia) graph-based disambiguation engine focused on the issue of name variance.

One of the tools we evaluated during the previous year (FREME[1]) was not included in this year's evaluation due to the rising number of errors returned by the tool. We have used the online versions of these tools for the evaluation, except for DBpedia Spotlight. Due to the fact that Spotlight currently returns a lot of errors, we have used the freely available Docker Spotlight container[2] for this evaluation. Since this container is used for the public Web service we consider the results to be equivalent. Surprisingly though, while there were still a number of errors and lower scores than usual, it was possible to compute evaluation scores using this container which might indicate that the Web service is using an older version of the container.

A script that integrates the open-source NIF converter[3] built by OpenTapioca developers was used for translating datasets between multiple schemas (e.g., DBpedia-Wikidata). For the purpose of translation, the datasets were not updated as we have considered that only the existing data should be translated. Only annotators that are able to perform NEL tasks have been currently integrated.

Table 1: Comparison of NEL systems performance on multiple corpora (m - micro; M - macro; p - precision; r - recall; $F1$ - F1).

Corpus	*System*	mP	mR	$mF1$	MP	MR	$MF1$
Reuters 128	AIDA	**0.65**	0.36	0.46	0.51	0.32	0.38
	Babelfy	0.62	0.35	0.45	0.53	0.33	0.39
	Spotlight	0.30	0.30	0.30	0.27	0.32	0.28
	Recognyze	0.62	**0.60**	**0.61**	**0.55**	**0.53**	**0.54**
RSS 500	AIDA	**0.64**	0.34	0.45	0.48	0.35	0.40
	Babelfy	0.63	0.34	0.44	0.46	0.34	0.38
	Spotlight	0.17	0.25	0.20	0.19	0.25	0.20
	Recognyze	0.63	**0.63**	**0.63**	**0.53**	**0.54**	**0.53**
OKE 2016	AIDA	0.64	0.41	0.50	0.66	0.43	0.51
	Babelfy	0.65	0.47	0.55	0.66	0.48	0.55
	Spotlight	0.52	0.27	0.35	0.30	0.25	0.26
	Recognyze	**0.86**	**0.69**	**0.77**	**0.80**	**0.65**	**0.72**

Since Orbis supports NIF, all publicly available datasets in this format are supported as well. The evaluations presented in this section draw upon the following gold standard datasets:

- *Reuters128* is part of the larger *N3 collection* [41] and contains texts with popular entities extracted from the classic Reuters corpora.
- *RSS500* is also a part of the larger *N3 collection* [41], but the texts were extracted from various blogs. The content is from the earlier part of the 2010s.
- *OKE2015* [35] and *OKE2016* [36] are two datasets used during the Open Knowledge Extraction Challenges at ESWC conferences. They contain short biographic sentences selected from Wikipedia abstracts. *OKE2016* includes all the texts from *OKE2015*, but adds a similar quantity of texts.

We present both micro and macro results for the evaluations. The micro results represent the weighted average scores, whereas the macro results represent the arithmetic mean of the per-class (e.g., type for NER/NEL) scores. Micro results are well-suited for presenting

[1] https://freme-project.github.io/
[2] https://github.com/dbpedia-spotlight/spotlight-docker
[3] https://github.com/wetneb/nifconverter

results for imbalanced classes (e.g., for NEL corpora this is typically the case as the number of examples that belong to the different entity types will almost always be imbalanced), whereas the macro-averages compute the metrics separately for each class and then take the averages. It is important to provide both metrics precisely because class distributions differ wildly between various documents or corpora.

Almost all tools have improved compared to last year, though in most cases we can see mostly improvement in precision. Recall seems to be difficult for most of the systems. AIDA and Recognyze show most improvements. The only tool that has not improved and in some cases showed worse results is Spotlight. This can also be due to the high number of errors thrown by the Web service or Docker container.

3.3.3. Recognyze Media Annotations

Besides core entity types like *Person*, *Organization* and *Location*, media-related document annotation also needs support for additional entity types like *Creative Work* [4] or *Work*. A wide array of entities that can be classified as creative works, from books and songs, through games, movies, TV Shows and entire media franchises are covered by this class. There is a lot of variation when it comes to the main attributes of this entity type as opposed to the core types (Person, Organization, Location), as in fact besides title, creator and a temporal attribute (e.g., first episode date, published in, etc.) the rest of the attributes will differ. The common thread is the fact that these creative works have all been published at some point through some platform (e.g., book, TV, movie), and while it can be argued that most should be modeled as their own entity types, it is still valuable to describe them as belonging to the same class. Inclusion of creative works can lead to a high number of false positives, as fictional characters can share names with real people or their own media franchises which encompass different works (e.g., *James Bond*, *Wolverine*); fictional characters might be based on real people (e.g., *Bohemian Rhapsody* movie or song, *Rocketman* movie or song).

We created a corpus[5] for testing the various limit scenarios that include creative works to help us fine-tune our media-related document annotations. We have started by collecting several sentences from the Wikipedia abstracts of 100 media entities. The initial set of entities contained books, TV shows, media companies, YouTube influencers and media franchises. Several entity types were annotated, include Person (PER), Organization (ORG), Location (LOC), Work (WORK) or Other (OTHER). The corpus was annotated by two annotators following an annotation guideline, and was later judged by a third annotator. The resulting corpus was exported into multiple formats, including csv and NIF.

All texts were collected from open-source repositories like Wikipedia, TV Tropes or Wiki News in order to enhance reproducibility. Currently the following partitions are available (we indicate the sources in parentheses), each with 100 documents of one to three sentences length:

- *General* (Wikipedia) partition contains short documents that reflect the core set of the available partitions from news and politics to franchises and influencers.
- *Franchises* (TV Tropes) set is focused on big multimedia franchises like *Marvel Cinematic Universe* or *Star Wars* and the creative works in various formats (movies, TV shows, books, video games) that support them.

[4] represented by `http://dbpedia.org/ontology/Work` (abbreviated as dbo:Work) in DBpedia or `https://schema.org/CreativeWork` in the schema.org vocabulary

[5] The corpus will be made publicly available in Q3 2020 through the GitHub page of MODUL Technology

- *RegionalTV* (TV Tropes) set contains texts about European TV Shows.
- *EuroFilm* (TV Tropes) is focused on classic and modern European films.
- *WebMedia* (TV Tropes) set was built around YouTube influencers therefore containing time-sensitive content.
- *News* (Wikinews) collects general interest news on a variety of topics.
- *Politics* (Wikinews) encompasses general politics news related to elections, political events (e.g., Syrian Conflict, Arab Spring) or war-related news.
- *Business* (Wikipedia) presents some data about corporations from tech, medical and media domains.

The annotation guideline and the rules we have used for annotating this corpora were first introduced in a conference publication that was already accepted at an ACL conference last year (RANLP 2019 [53]).

The following annotation styles were considered for the current evaluation (here illustrated based on the annotation of the text snippet *Vienna, VA*):

1. *ØMIN* disregards overlapping entities and extracts the minimum number of entities: $m^{dbr:Vienna,\_Virginia}_{\text{[Vienna, VA]}}$, i.e. links the snippet to the *Vienna, Virginia* DBpedia entity.
2. The annotation style *ØMAX*, in contrast, extracts the maximum number of entities from a given text snippet: $m^{dbr:Vienna,\_Virginia}_{\text{[Vienna]}}, m^{dbr:Virginia}_{\text{[VA]}}$
3. The style *OMAX* allows for overlaps and, again, will aim to extract the maximum number of entities whenever possible: $m^{dbr:Vienna,\_Virginia}_{\text{[Vienna, VA]}}, m^{dbr:Virginia}_{\text{[VA]}}$

The presented rules only consider borderline cases, even though combinations of them can also be used within a corpus. A corpus which would not apply the OMAX rule, for example, might lose the extended reference to *Sir Patrick Stewart OBE* and only return *Patrick Stewart* or end up removing the references to the actor's titles (e.g., *Sir*, *OBE*). We consider OMAX annotation rule to be the best, as it essentially merges the other annotation styles.

Table 2: Comparison of NEL systems performance on a corpora with multiple lenses (m - micro; M - macro; p - precision; r - recall; $F1$ - F1).

Corpus	*System*	mP	mR	$mF1$	MP	MR	$MF1$
Mediacorpus100 ØMIN	AIDA	0.47	0.48	0.47	0.43	0.48	0.43
	Babelfy	0.33	0.35	0.34	0.37	0.37	0.35
	Spotlight	0.53	0.43	0.48	0.35	0.42	0.37
	Recognyze	**0.61**	**0.52**	**0.56**	**0.52**	**0.50**	**0.51**
Mediacorpus100 ØMAX	AIDA	0.49	0.48	0.49	0.45	0.48	0.44
	Babelfy	0.36	0.36	0.36	0.39	0.37	0.35
	Spotlight	0.55	0.43	0.48	0.35	0.40	0.36
	Recognyze	**0.62**	**0.54**	**0.58**	**0.55**	**0.52**	**0.53**
Mediacorpus100 OMAX	AIDA	0.49	0.48	0.49	0.45	0.48	0.44
	Babelfy	0.36	0.36	0.36	0.39	0.37	0.36
	Spotlight	0.51	0.57	0.54	0.51	**0.58**	0.52
	Recognyze	**0.65**	**0.61**	**0.64**	**0.61**	0.57	**0.59**

As it can be seen, these rules have only impacted three of the considered tools, AIDA being the only one whose results do not drastically change when considering these changes. Rest of

the tools do seem to perform better and better when considering these styles in succession, with Spotlight and Recognyze gaining up to 4%. Interesting to note, but the rules seem to improve the recall of DBpedia Spotlight and Recognyze in all cases, whereas precision is not impacted for OMAX styles for Spotlight. There might be a need for multiple evaluations in a future publication to establish the full impact of these guidelines, but since such annotation styles can automatically be generated from any dataset following the outlined rules, they are definitely worth investigating.

The evaluations show that our Recognyze-NG system repeatedly performs more accurately compared to other state of the art competitors in most types of evaluation, and consistently outperforms them when annotating entities of type Creative Work.

The improvements that were done while testing Recognyze with various KGs are described in an open conference publication[54]. The different annotation lenses were discussed at length in a paper accepted at an ACL conference [52]. A journal publication about Orbis was also submitted at a high-impact journal and is in the process of being reviewed.

4. Concept-Based Video Abstractions

4.1. Video Fragmentation

4.1.1. Updates from Previous Report

The problem of video fragmentation and the relevant literature survey are presented in detail in Section 4.1 of D1.1 [31] and Section 4.1 of D1.2 [32] of ReTV. In this section we discuss and address issues that came up after the delivery of D1.2, i.e., in months M21 to M30; specifically, the performance optimization of video fragmentation, the adjustment of the ReTV Video Analysis (VA) service to be able to ingest large video files, and a new video fragmentation module implemented to support a "Sandmännchen and Friends" scenario after discussions with a ReTV content partner.

Performance Optimization

As discussed in Section 4.1.1 of D1.2 the video fragmentation module that reads the video, decodes it and proceeds to segment it to scenes, shots and sub-shots is the most demanding stage in terms of time complexity. Our efforts on optimizing this stage in the past months include the following steps:

- Integrated a DCNN-based shot and scene segmentation (see Section 4.1 of D1.2).
- Employed a fast video read-and-decode workflow by applying a multi-thread scheme, where one thread reads the next batch of frames from the disk while a different one decodes the previously read batch of frames.
- Re-implemented the sub-shot segmentation using the same programming language that the service's script uses, eliminating the need to call an external C++ executable and the use of intermediate files. The re-implemented module was evaluated on the same datasets as the original method ([1]) and was found to have the exact same performance, being much more efficient.
- Re-designed the processing workflow in a way that shot segmentation is performed on-the-fly during video reading. More specifically, the shot transition prediction probabilities of the shot segmentation DCNN are calculated for overlapping batches of frames during the video reading. This ensures the maximum possible concurrent utilization of CPU and GPU.
- Similarly, re-designed the workflow in way that sub-shot segmentation is performed on-the-fly. The optical flow information (for details on the method see [1]) is extracted from overlapping batches of frames during the video reading.
- Re-designed the video reading framework to keep all frames available in the computer's working memory. After sub-shot segmentation only the sub-shot key-frames are kept while the rest are discarded. This way, all needed video frames for the next stages (i.e. concept detection, object detection, etc.) are readily available without having to re-read the video or the key-frames written on disk by the sub-shot segmentation sub-module.

The performance optimization results of the above described steps are reported in Section 4.1.2.

VA service modifications for ingesting large video files

The outcome of the aforementioned optimization procedure, is a framework where working memory (i.e. RAM) is exhaustively utilized in favor of fast execution times. As a consequence,

there is a physical limit on the duration of videos that can be ingested. Specifically, this limit was found to be 30 minutes based on the available RAM of the server that hosts the VA service. At first this did not seem to be a problem, since most of the video content that the VA service ingests is sent by GENISTAT in 15 minutes-long video segments. However, a content partner discussed the possibility to analyse movie files, i.e. videos with duration over two hours.

To be able to overcome this physical limit on the videos duration, we implemented a memory-mapping scheme; large arrays are written on disk but cached in RAM. This makes the process of VA approximately 115% slower but this way we are able to ingest video files of more than 160 minutes duration. The memory mapping scheme is automatically used only when a video with duration over 30 minutes is submitted. It is worth noting that the video analysis is a necessary step to be able to generate video summaries for such large video files - according to the final ReTV architecture, features extracted by VA are prerequisites for the Video Summarization service of WP3. Therefore, the ability to ingest and analyse large video files implies that we are also able to generate summaries of such large video files.

Video structure for "Sandmännchen and Friends" use-case

In discussions with the RBB content partner, CERTH was notified about the "Sandmännchen and Friends" scenario in which episodes of children's series have to be analyzed and the features extracted would later be utilized by WP6 of ReTV. There was a request to implement a support signal for the "Sandmännchen and Friends" scenario, namely, to detect the structure of a "Sandmännchen and Friends" episode. The "Sandmännchen and Friends" episodes have three parts:

1. The introductory part, where the Sandmänn arrives (in a different vehicle every time) and enters a room with children, and starts narrating his story.
2. The main part of the episode is the story where the Sandmänn is not visible, and
3. The closing part, when Sandmänn has finished his narration and he is leaving.

After visual inspection of many "Sandmännchen and Friends" episodes we decided to detect the "intro" transition (i.e, transition from the introductory part to the main story) and the "outro" transition of an episode (i.e, transition from the main story to the closing part of the episode). In most cases, the frames around the "intro" and "outro" transitions contain a camera zooming in and out from a screen, respectively. The screen is different every time, sometimes being a TV screen, other times being just a projection on wall. The zooming is accompanied with a fading transition, where in most cases the camera zooming fading out to a white frame.

We trained a Random Forest classifier on various sets of features already extracted by the VA service to classify video frames into two classes: "normal frame" and "transition frame". We are not relying solely on this frame-level prediction (i.e., whether a frame was correctly classified to belong to a transition) but we also calculate a video-level prediction (i.e., the 2 shots of the video where the 'intro" and "outro" transitions happen). In the video-level inference we employ some additional domain rules, specifically:

- For a frame to be considered a "transition frame", besides having a high inferred prediction probability from the Random Forest classifier, must also belong to either the first or the last 1/3 of the video. This rule was employed since the main story part on all analysed "Sandmännchen and Friends" episodes was the largest part and always in the middle of the video.

- For a frame to be considered a "transition frame", it must additionally have a temporal distance of four seconds at most from the end of a shot. We employed this rule since the largest transition was observed to be four seconds and the 'intro" and "outro" transitions are always marked as a shot change by the shot segmentation module.

After the application of these additional domain rules we select the shot that contains the highest ranked "transition frame" and belongs to the first $1/3$ portion of the video as the last shot of the introductory part. Consequently, we select the shot that contains the highest ranked "transition frame" and belongs to the last $1/3$ portion of the video as the last shot of the main story.

4.1.2. Results

In order to measure the speedup after the performance optimization steps, we compiled a small test dataset by selecting 21 random YouTube videos, of various content, and a total playing time of 123 minutes. Since many of the stages of the VA service are now interleaved and it is not possible to measure e.g. the exact process time of the video fragmentation module alone, we choose to compute the ratio of the whole processing time to the total video duration for each tested video. Table 3 shows the results of these tests. It should be stressed that the results of this table include all the optimization efforts, discussed in Section 4.1.1 as well as in Sections 4.2 and 5.3. We observe the significant speedup and a greater consistency, i.e., smaller range of processing times.

Table 3: Performance optimization results.

	Ratio of session processing time to video duration (mean ± std.)	
	Brand detection enabled	Brand detection disabled
Video Analysis component before optimization	1.31 ± 0.22	1.05 ± 0.19
Video Analysis component after optimization	0.79 ± 0.11	0.53 ± 0.08

Regarding the "Sandmännchen and Friends" episode structure segmentation, we compiled a training dataset with 50 "Sandmännchen and Friends" episodes, by manually annotating the structure of each episode. We also annotated 30 "Sandmännchen and Friends" videos to create a testing dataset, but this time the list of videos was compiled by RBB. We tested various subsets of features to train our Random Forest classifier on. In Table 4 we report the results of our the final setup, which uses 6 different features, namely a) frame whiteness, b) frame blankness, c) frame blackness, d) frame blurriness, e) frame's Egde Change Rate (ECR) measure, and f) ECR of the shot that the frame belongs to. Using the final setup, we achieve 88.54% frame-level F-score, while when employing the additional domain rules we reach a 91.67% F-score at video-level.

4.2. Concept-Based Annotation

During this period, we focused on the performance optimization of the concept-based annotation module of the VA service. Our efforts can be divided into two categories: a) algorithmic performance optimization and b) implementation-related performance optimization. As part of the algorithmic performance optimization effort, we worked on compacting DCNNs using

Table 4: Confusion matrix for "Sandmännchen and Friends" episode structure identification.

	Normal frame	*Transition frame*
Normal frame	0.94	0.06
Transition frame	0.01	0.99

pruning techniques. In the context of implementation-related performance optimization, we re-designed parts of the concept annotation for YT8M concepts to limit the use of intermediate files, and we investigated ways to re-train the SIN concept pool model using a modern and more efficient deep learning framework. Finally, besides the optimization effort, we employed a new model for the main character identification in "Sandmännchen and Friends" episodes in the content of designing signals to aid the "Sandmännchen and Friends" scenario.

4.2.1. Updated Problem Statement and State of the Art

Deep neural networks (DNNs) have shown outstanding classification performance in a variety of application domains. However, limitations in the computational capabilities of resource-limited devices such as IoT and mobile devices, inhibit the use of top-performing DNNs in these areas. Moreover, the computational time required for the execution of powerful DNNs is prohibitive for real-time applications. Many approaches have been proposed to reduce the space requirements and accelerate large DNNs, including quantization, low-rank approximations, and other [37, 4].

Pruning is recently getting increasing attention due to the fact that it can be used in combination with most of the other compression and acceleration methods. It typically consists of an importance estimation criterion and a pruning strategy. For instance, in [27], the l_1 norm filter selection criterion is utilized with different pruning rates per layer. In [33], a Taylor expansion-based criterion is proposed to approximate the accuracy loss of pruned feature maps. In [18], a LASSO regression framework is utilized. In [8], low-cost collaborative layers are used to prune filters with zero responses after the ReLU activation. A hybrid algorithm is introduced in [10], combining low rank approximation, quantization and pruning. In [56], filter- and shape-wise scaling factors are used for filter weakening and pruning. In [16], filters are pruned iteratively along epochs using an l_2 norm-based criterion. The above method is extended in [15] using an asymptotic filter pruning schedule. In [55], intrinsic sparse structures and a weight sparcification technique are used to prune LSTM cells in recurrent neural networks (RNNs). In [28], a trained DCNN and a k-means-based criterion are utilized for filter representation and pruning. In [17], the geometric median (GM) is used to design a criterion for selecting the filters with the most replaceable contribution.

As described above, most pruning approaches in the literature utilize energy preserving-based criteria for filter selection, which may be suboptimal from the perspective of classification [9, 12]. To this end, instead of energy preservation measures, we consider the between-class scatter matrix, which is a popular class-separability measure utilized for extracting the most effective features for classification, and combine it with the GM-based criterion presented in [17]. Furthermore, inspired from [12] we adapt and extend the approach of [15] so that both the pruning rate and filter weights' scaling factor asymptotically approximate their target values. In this way, selected filters are not pruned immediately, but instead are "squeezed" towards zero

in small fractional steps, preserving more effectively the capacity of the network and yielding a more stable procedure. The proposed method is evaluated for DCNN pruning in three datasets, namely, CIFAR-10, Google speech commands (GSC) and ImageNet32 (a downsampled version of ILSVRC-2012). Moreover, a variant of the proposed approach is evaluated for RNN pruning using the YouTube8M dataset.

4.2.2. Updated Deep Learning Approach for the TRECVID SIN Concepts

In the Section 4.2.3 of D1.1 we described our method for training a model to annotate video frames with the TRECVID SIN concepts [38]. This was implemented using the Caffe Deep Learning (DL) framework (see Section 4.2.4 of D1.1) and employed in a C++ standalone executable. In order to replace this, now outdated, DL framework and eliminate the need to call an external executable and the consequent use of large intermediate files we decided to retrain a model on the same concept pool, this time using a modern DL framework allowing the whole procedure to be implemented in the same programming language that the server script uses and thus exploit the performance optimization steps discussed in Section 4.1.1.

4.2.3. Pruning Techniques for Neural Networks

DCNN pruning

Consider a DCNN $\mathcal{W}$ with l convolutional layers and c filters, each filter denoted as:

$$\mathcal{V}^{(i,j)} \in \mathbb{R}^{k\times k\times c_{i-1}}, i = 1,\ldots,l,\ j = 1,\ldots,c_i, \tag{1}$$

where i is the layer index, j and c_i are the filter index[6] and number of filters in the ith layer, respectively, k is the kernel size and $c = \sum_{i=1}^{l} c_i$. Suppose an annotated training dataset $\mathfrak{D}$ of n observations belonging to m disjoint classes

$$\mathfrak{D} = \{(\mathcal{X}_1^{(0)}, \mathbf{y}_1), \ldots, (\mathcal{X}_n^{(0)}, \mathbf{y}_n)\}, \tag{2}$$

where, $\mathcal{X}_\kappa^{(0)} = [\mathcal{X}_\kappa^{(0,1)}, \ldots, \mathcal{X}_\kappa^{(0,c_0)}] \in \mathbb{R}^{h_0\times w_0\times c_0}$ is the tensor representing the κth observation, h_0, w_0, c_0 are the tensor's height, width and number of channels, respectively, $\mathcal{X}_\kappa^{(0,j)} \in \mathbb{R}^{h_0\times w_0}$ is the slice of $\mathcal{X}_\kappa^{(0)}$ corresponding to channel j, $\mathbf{y}_\kappa \in \mathbb{R}^m$ is the class indicator vector, i.e. its pth element is one if $\mathcal{X}_\kappa^{(0)}$ belongs to class υ_p and zero otherwise, and υ_p denotes the pth class. In modern CNN architectures, the feature map $\mathcal{X}_\kappa^{(i,j)} \in \mathbb{R}^{h_i\times w_i}$ corresponding to filter (i,j) and observation κ is typically computed using

$$\mathcal{X}_\kappa^{(i,j)} = \mathrm{ReLU}(\mathrm{BN}(\sum_{\tau=1}^{c_{i-1}} \mathcal{X}_\kappa^{(i-1,\tau)} * \mathcal{V}_{:,:,\tau}^{(i,j)})), \tag{3}$$

where, BN(), ReLU() are the batch normalization and rectified linear unit operators, respectively, h_i, w_i are the height and width of $\mathcal{X}_\kappa^{(i,j)}$, $\mathcal{V}_{:,:,\tau}^{(i,j)}$ is the slice of (i,j) filter corresponding to the τth input channel and $*$ is the two-dimensional convolution operator. Given a target pruning rate θ and an importance estimation mapping η, in most approaches the set $\mathsf{F}^{(i)}$ of filters to prune in layer i is computed using the following criterion

$$\mathsf{F}^{(i)} = \mathrm{argmin}(\boldsymbol{\eta}^{(i)}, \theta c_i), \tag{4}$$

[6] Note that in the context of DCNNs, j is also used as channel index of feature maps, which is a common practice in the literature.

Algorithm 1: Fractional step discriminant pruning

Input: $\mathcal{W}$, $\mathfrak{D}$ (2), δ, ε, θ (10), (11), $\hat{\vartheta}_f$ (13)
Output: Pruned $\mathcal{W}$

Compute α, β, γ in (10) using (12)
for $\iota \leftarrow 1$ *to* ε **do**
  Update CNN parameters using training set $\mathfrak{D}$ (2)
  Compute pruning rate ϑ_ι (10) and scaling ζ_ι (11)
  Compute pruning rates $\hat{\vartheta}_\iota$ (13), $\tilde{\vartheta}_\iota$ (14) for the discriminant and GM-based criterion
  for $i \leftarrow 1$ *to* l **do**
    for $j \leftarrow 1$ *to* c_i **do**
      Compute discriminant score $\hat{\eta}^{(i,j)}$ (5)
    Form set $\mathsf{F}^{(i)}$ (4) with the indexes of the $\hat{\vartheta}_\iota c_i$ filters with smaller $\hat{\eta}^{(i,j)}$
    for $j \leftarrow 1$ *to* c_i; $(i,j) \notin \mathsf{F}^{(i)}$ **do**
      Compute GM-based score $\tilde{\eta}^{(i,j)}$ (9)
    Add to $\mathsf{F}^{(i)}$ (4) the indexes of the $\tilde{\vartheta}_\iota c_i$ filters with smaller $\tilde{\eta}^{(i,j)}$
    Scale the weights of the filters in $\mathsf{F}^{(i)}$ using ζ_ι
Prune the filters in $\mathsf{F}^{(i)}$ (4) $\forall i$

where, $\boldsymbol{\eta}^{(i)} = [\eta^{(i,1)}, \ldots, \eta^{(i,c_i)}]^T$ is a vector whose component $\eta^{(i,j)}$ is the output of η associated with the filter (i,j) and the use of the argmin$(,)$ vector operator above returns the indexes of the θc_i smaller components in $\boldsymbol{\eta}^{(i)}$.

Filter importance estimation mapping

Class-separability measures have shown superior performance in comparison to energy preserving criteria in many classification problems [9, 12]. To this end, we resort to the trace of the between-class scatter matrix for designing a suitable criterion for network pruning. For simplicity of notation the indexes (i,j) are dropped in this section as the analysis in the following is valid for any filter in the network. The feature maps, $\mathcal{X}_\kappa \in \mathbb{R}^{h \times w}, \kappa = 1, \ldots, n$, associated with a specific filter can be vectorized and stacked column-wise to form a matrix $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}^{f \times n}$, where $\mathbf{x}_\kappa = \text{vec}(\mathcal{X}_\kappa)$, $\mathbf{x}_\kappa \in \mathbb{R}^f$, $f = hw$, vec() is the vectorization operator, and h, w are the height and width of the feature maps. The filter discriminant score is then computed using

$$\hat{\eta} = \text{tr}(\mathbf{S}), \tag{5}$$

where, tr() is the matrix trace operator, $\mathbf{S}$ is a variant of the between-class scatter matrix defined as

$$\mathbf{S} = \sum_{p=1}^{m-1} \sum_{q=p+1}^{m} (\boldsymbol{\mu}_p - \boldsymbol{\mu}_q)(\boldsymbol{\mu}_p - \boldsymbol{\mu}_q)^T, \tag{6}$$

and $\boldsymbol{\mu}_p = \frac{1}{n_p} \sum_{\mathbf{x}_\kappa \in \upsilon_p} \mathbf{x}_\kappa$, n_p, are the estimated mean vector and cardinality of class υ_p. The computation of $\mathbf{S}$ using (6) is susceptible to memory restrictions and does not fully utilize the parallelization capabilities of modern GPUs. To this end, the matrix $\mathbf{M}$ of class means can be factorized as

$$\mathbf{M} = [\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_m] = \mathbf{XRA}, \tag{7}$$

where, $\mathbf{R} = [\mathbf{y}_1, \ldots, \mathbf{y}_n]^T \in \mathbb{R}^{n \times m}$ is the class indicator matrix and $\mathbf{A} \in \mathbb{R}^{m \times m}$ is a diagonal matrix defined as $\mathbf{A} = \text{diag}(\frac{1}{n_1}, \ldots, \frac{1}{n_m})$. For large-scale datasets it is infeasible to load the whole matrix $\mathbf{X}$ into the memory, and the same is true for $\mathbf{R}$ when the number of classes is large. Instead, assuming that these matrices are partitioned to t blocks, i.e., $\mathbf{X} = [\mathbf{X}_1, \ldots, \mathbf{X}_t]$, $\mathbf{R} = [\mathbf{R}_1^T, \ldots, \mathbf{R}_t^T]^T$, where $\mathbf{R}_j$ is the indicator matrix corresponding to $\mathbf{X}_j$, the matrix product $\mathbf{XR}$ can be computed very efficiently in the GPU using $\mathbf{XR} = \sum_{j=1}^{t} \mathbf{X}_j \mathbf{R}_j$. Finally, $\mathbf{S}$ can be factorized as follows

$$\begin{aligned}
\mathbf{S} &= \sum_{p=1}^{m-1} \sum_{q=p+1}^{m} (\boldsymbol{\mu}_p \boldsymbol{\mu}_p^T + \boldsymbol{\mu}_q \boldsymbol{\mu}_q^T - \boldsymbol{\mu}_p \boldsymbol{\mu}_q^T - \boldsymbol{\mu}_q \boldsymbol{\mu}_p^T) \\
&= (m-1) \sum_{p=1}^{m} \boldsymbol{\mu}_p \boldsymbol{\mu}_p^T - \sum_{p=1}^{m-1} \boldsymbol{\mu}_p (\sum_{q=p+1}^{m} \boldsymbol{\mu}_q^T) \\
&\quad - \sum_{p=1}^{m-1} (\sum_{q=p+1}^{m} \boldsymbol{\mu}_q) \boldsymbol{\mu}_p^T \\
&= (m-1)\mathbf{M}\mathbf{M}^T - \mathbf{M}(\mathbf{J} - \mathbf{I})\mathbf{M}^T \\
&= \mathbf{M}(m\mathbf{I} - \mathbf{J})\mathbf{M}^T, \qquad (8)
\end{aligned}$$

where, $\mathbf{I}$ and $\mathbf{J}$ are the $m \times m$ identity and all-one matrices respectively. Using the last expression and provided that the matrix of class means has been derived, the between-class scatter matrix can be computed very efficiently in the GPU.

Geometric median-based criterion

In more challenging problems, the application of the discriminant criterion with a relatively large pruning rate may eliminate filters with small but still important discriminant information, harming the classification performance of the network (as for instance we have seen in the experimental evaluation of ImageNet32 in Section 4.2.7). To this end, we select a fraction of the filters using the discriminant criterion in (5), and from the remaining filters another fraction is selected exploiting a GM-based scoring function defined as [17]

$$\tilde{\eta}^{(i,j)} = \sum_{o=1}^{c_i} ||\mathbf{v}^{(i,j)} - \mathbf{v}^{(i,o)}||_2, \qquad (9)$$

where, $\tilde{\eta}^{(i,j)}$ is the importance score for filter (i,j) and $\mathbf{v}^{(i,j)} = \text{vec}(\mathcal{V}^{(i,j)})$.

Fractional step pruning strategy

Most recent approaches select and prune θc filters at every epoch of the pruning procedure. When the pruning rate is large, following the above strategy will reduce abruptly the network capacity and at the same time discard a large amount of discriminant information [16, 15]. To alleviate these drawbacks, an asymptotic soft filter pruning approach was presented in [15]. Inspired from [12], here we extend the pruning strategy of [15] so that the parameters of the selected filters are not set to zero at every epoch, but on the contrary are multiplied with a scaling factor that decreases from one to zero along the training procedure. Thus, the selected filters are compressed in small fractional steps towards zero and the capacity of the network decreases smoothly. In more detail, assuming that ε, θ are the total epochs and desired pruning rate, respectively, the pruning rate ϑ_i and scaling factor ζ_i at epoch i are computed using the

following asymptotic schedule

$$\vartheta_\iota = \alpha \exp(-\beta\iota) + \gamma, \tag{10}$$

$$\zeta_\iota = 1 - \frac{\vartheta_\iota}{\theta}, \tag{11}$$

where, $\vartheta_\varepsilon = \theta$, and α, β, γ are the parameters of the asymptotic function. Similarly to [15], the estimation of these three parameters is performed using the following three points for the epoch and pruning rate tuple $\{\iota, \theta_\iota\}$

$$\{0,0\},\ \{\delta\varepsilon,\ \frac{3}{4}\theta\},\ \{\varepsilon,\theta\}, \tag{12}$$

where $\delta \in (0,1)$. Moreover, the individual pruning rates at each epoch, $\hat{\vartheta}_\iota$, $\tilde{\vartheta}_\iota$ for the discriminant and GM-based criterion, respectively, are computed as follows

$$\hat{\vartheta}_\iota = \min(\vartheta_\iota, \hat{\vartheta}_f), \tag{13}$$

$$\tilde{\vartheta}_\iota = \vartheta_\iota - \hat{\vartheta}_\iota, \tag{14}$$

where $\hat{\vartheta}_f$ is a parameter denoting the final pruning rate associated with the discriminant criterion. We observe that the sum of the two individual rates always equals to the total rate at each epoch ($\hat{\vartheta}_\iota + \tilde{\vartheta}_\iota = \vartheta_\iota$) and the pruning rate $\hat{\vartheta}_\iota$ associated with the discriminant criterion (5) is never larger than $\hat{\vartheta}_f$, ensuring that filters with small but possible significant information are not pruned.

The pseudocode of the proposed method is shown in Algorithm 1. We should note that the only input parameters of the method are δ (12) and $\hat{\vartheta}_f$ (13), which based on the related literature and our experimental evaluation in Section 4.2.7, are typically set to $\frac{1}{8}$ and 10%, respectively (e.g. see ASFP [15] and FPGM [17]).

LSTM pruning

A variant of the proposed technique can be used for pruning LSTM layers in Recurent Neural Networks (RNNs). Suppose a training dataset contaning n vector sequences and C classes:

$$\mathcal{G} = \{(\mathbf{A}_1, \mathbf{y}_1), \ldots, (\mathbf{A}_n, \mathbf{y}_n)\}, \tag{15}$$

where, $\mathbf{A}_\kappa = [\mathbf{a}_{\kappa,1}, \ldots, \mathbf{a}_{\kappa,T_\kappa}] \in \mathbb{R}^{f \times T_\kappa}$ is the κth vector sequence with length T_κ and $\mathbf{y}_\kappa = [\mathbf{y}_{\kappa,1}, \ldots, \mathbf{y}_{\kappa,C}]^T$ is the respective class indicator vector. Let $\mathbf{W}_1^{(i)}, \mathbf{W}_2^{(i)}, \mathbf{W}_3^{(i)}, \mathbf{W}_4^{(i)} \in \mathbb{R}^{h_i \times f_i}$, $\mathbf{U}_1^{(i)}, \mathbf{U}_2^{(i)}, \mathbf{U}_3^{(i)}, \mathbf{U}_4^{(i)} \in \mathbb{R}^{h_i \times h_i}$, $\mathbf{b}_1^{(i)}, \mathbf{b}_2^{(i)}, \mathbf{b}_3^{(i)}, \mathbf{b}_4^{(i)} \in \mathbb{R}^{h_i}$, be the parameters of the ith LSTM layer, such that [19]

$$\begin{aligned}
\mathbf{g}_{1,t}^{(i)} &= \sigma(\mathbf{W}_1^{(i)} \mathbf{a}_{\kappa,t}^{(i)} + \mathbf{U}_1^{(i)} \mathbf{z}_{\kappa,t-1}^{(i)} + \mathbf{b}_1^{(i)}), \\
\mathbf{g}_{2,t}^{(i)} &= \sigma(\mathbf{W}_2^{(i)} \mathbf{a}_{\kappa,t}^{(i)} + \mathbf{U}_2^{(i)} \mathbf{z}_{\kappa,t-1}^{(i)} + \mathbf{b}_2^{(i)}), \\
\mathbf{g}_{3,t}^{(i)} &= \sigma(\mathbf{W}_3^{(i)} \mathbf{a}_{\kappa,t}^{(i)} + \mathbf{U}_3^{(i)} \mathbf{z}_{\kappa,t-1}^{(i)} + \mathbf{b}_3^{(i)}), \\
\mathbf{g}_{4,t}^{(i)} &= tanh(\mathbf{W}_4^{(i)} \mathbf{a}_{\kappa,t}^{(i)} + \mathbf{U}_4^{(i)} \mathbf{z}_{\kappa,t-1}^{(i)} + \mathbf{b}_4^{(i)}), \\
\mathbf{g}_{5,t}^{(i)} &= \mathbf{g}_{2,t}^{(i)} \odot \mathbf{g}_{5,t-1}^{(i)} + \mathbf{g}_{1,t}^{(i)} \odot \mathbf{g}_{4,t}^{(i)}, \\
\mathbf{z}_{\kappa,t}^{(i)} &= \mathbf{g}_{3,t}^{(i)} \odot tanh(\mathbf{g}_{5,t}^{(i)})
\end{aligned}$$

where, $\mathbf{a}_{\kappa,t}^{(i)} \in \mathbb{R}^{f_i}$ is the input vector corresponding to the κth training observation, $\mathbf{g}_{1,t}^{(i)}$, $\mathbf{g}_{2,t}^{(i)}$, $\mathbf{g}_{3,t}^{(i)}$, $\mathbf{g}_{5,t}^{(i)}$, $\mathbf{z}_{\kappa,t}^{(i)}$ are the input, forget, output, cell state and hidden state vector updates,

respectively, h_i is the number of cells in the ith layer and $\odot$ is element-wise multiplication. The weight matrices of the ith layer can then be stacked to form a matrix $\mathbf{V}^{(i)} \in \mathbb{R}^{h_i \times 4(h_i+f_i)}$ defined as

$$\mathbf{V}^{(i)} = [\mathbf{W}_1^{(i)}, \mathbf{W}_2^{(i)}, \mathbf{W}_3^{(i)}, \mathbf{W}_4^{(i)}, \mathbf{U}_1^{(i)}, \mathbf{U}_2^{(i)}, \mathbf{U}_3^{(i)}, \mathbf{U}_4^{(i)}]. \quad (16)$$

The jth row of the above matrix, denoted hereafter as $\mathbf{v}^{(i,j)}$, is directly related with the jth LSTM cell in the ith layer. This allows us to utilize the GM-based scoring function for filter importance estimation (see eqs. (9), (4)) and apply an iterative scheme for LSTM pruning, as explained above for the DCNN pruning approach.

4.2.4. Character Identification for "Sandmännchen and Friends"

In discussions with the RBB content partner, we came to the conclusion that another needed signal to support the "Sandmännchen and Friends" scenario was the main character identification in the episodes. In Section 4.1.1 we mentioned that these episodes consist of three segments: a) the introductory part, b) the main story, and c) the closing part. The Sandmänn appears in the introduction and closing part while the main story deals with a different character each time. We trained a DCNN model to be able to detect the character of this main story. The first model could detect the following characters: 1) Fuchs&Elster, 2) Jan&Henry, 3) Kalli, 4) Konig, 5) Moffels, 6) Rita, 7) Sandmänn, 8) no character (background class). In the later months, after a first round of tests and discussions with RBB regarding the updated requirements for the "Sandmännchen and Friends" scenario, we trained a new model on a richer training dataset. The new model can detect 12 characters: 1) HerrFuchs&FrauElster, 2) Jan&Henry, 3) Kalli, 4) KleineKonig, 5) KleineRabeSocke, 6) Luzi&Moffels, 7) MeineSchmusedecke, 8) Pittiplatsch&Schnatterinchen&Moppi, 9) Plumps, 10) Pondorondo, 11) Rita&Krokodil, 12) Sandmänn. In Section 4.2.7 we report the evaluation results of the latest "Sandmännchen and Friends" character identification model.

As in the case of "Sandmännchen and Friends" episodes structure identification, discussed in Section 4.1.1, we do not rely solely on the frame-level character predictions but we also calculate a video-level prediction in which we perform majority voting over the frame-level predictions, since a "Sandmännchen and Friends" episode deals with a single character in its main story part.

Our models were trained and evaluated on a dataset curated by RBB. The specifications of this dataset are reported in Table 5. The training dataset was provided in the form of a set of frames while the testing dataset was provided as videos to evaluate the whole character identification process, including the video-level character inference.

Table 5: "Sandmännchen and Friends" character identification dataset information.

	Train dataset	*Test dataset*
Videos	N/A	35
Shots	N/A	1453
Frames analyzed	14375	238535
Total duration	N/A	9541 seconds

4.2.5. ReTV Method for Content Type Classification

We designed a Content-type Classifier that can classify video shots in the following classes: "content:news", "content:sports", "content:other", "ad" or "promo". The "content" classes

includes any kind of TV show ("content:other"), TV news program ("content:news") or TV sports programs ("content:sports"). "ad" labels shots with a advertisement about a brand. Finally "promo" refers to shots containing an advertisement regarding a TV show of the same channel that will be transmitted at a later time.

Using the ad-classifier (presented in Section 5.3 of D1.2) we can label shots as: content, ad or promo. To detect shots with news content, we utilized the annotations of certain news-related concepts from the SIN concept pool, namely 1) Studio_With_Anchorperson, 2) Female_Anchor, 3) Male_Anchor, 4) News_Studio, and 5) News. These form our news-related concept set. To detect shots with sports content we trained a DCNN model on the Sports dataset[7], a dataset created by downloading photos from Google Images for 22 sports, which contains in total over 14.500 images. The resulting model can annotate a frame with the following labels: 1) badminton, 2) baseball, 3) basketball, 4) boxing, 5) chess, 6) cricket, 7) fencing, 8) football, 9) formula1, 10) gymnastics, 11) hockey, 12) ice_hockey, 13) kabaddi, 14) motogp, 15) shooting, 16) swimming, 17) table_tennis, 18) tennis, 19) volleyball, 20) weight_lifting, 21) wrestling, 22) wwe, and 23) no_sport. The first 22 concepts of the sports DCNN model form our sports-related concept set.

Our Content-type Classifier works as follows: First, we employ the video analysis so that shot segmentation results as well as concept annotations are available. We employ the ad-detector of D1.2 to label each shot with a "ad" or "promo" or "content". For the shots that where labeled as "content" we proceed to calculate the average probability of the news-related concept set, the sports-related concept set as well as the "no_sport" concept over all shot frames. If one averaged concept probability from the sports-related concept set is larger than the averaged "no_sport" concept probability, then the shot is labeled as "content:sports". Otherwise, if the sum of the averaged probabilities of the news-related set is larger than a pre-defined threshold, then the shot is labeled as "content:news". In any other case the shot is labeled as "content:other".

4.2.6. Implementation Details and Use

To train the new models for the SIN concept pool and the new models for character identification, we used the Keras[8] open-source neural-network library version 2.3.1, with the TensorFlow backend[9] version 1.14. All experiments where conducted on an Intel i5 9600K, PC with 32 GB RAM, running Ubuntu 18, equipped with an Nvidia GeForce GPU (RTX 2080 Ti).

For the experiments on pruning techniques we used the TensorFlow neural-network library version version 1.14, and the experimental evaluation was performed on an Intel i7 3770K PC with 32 GB RAM, running Windows 10, equipped with an Nvidia GeForce GPU (GTX 1080 Ti).

4.2.7. Results

Re-training models for the SIN concept pool

The TRECVID SIN training dataset consists of over 33 thousand videos. Along with the original videos, multiple keyframes per shot are provided. We choose to use the provided shot keyframes instead of the videos and employed three approaches:

[7] https://www.pyimagesearch.com/2019/07/15/video-classification-with-keras-and-deep-learning/
[8] `https://keras.io/`
[9] `https://www.tensorflow.org/`

- B1_1KFPS: We trained an EfficientNetB1 DCNN model on the middle keyframe of each shot.
- B3_3KFPS: We trained an EfficientNetB3 DCNN model on three temporally equally-spaced keyframes from each shot (for shots that multiple keyframes are provided).
- B3_1KFPS: We trained an EfficientNetB3 DCNN model on the middle keyframe of each shot.

We trained our models for 100 epochs, except for the B3_3KFPS approach for which we stopped the training procedure at an early stage due to the poor results of the training accuracy reported after each epoch. For the remaining approaches (i.e. B1_1KFPS and B3_1KFPS), we selected to evaluate the model snapshots of the epoch right before the model's loss reached a plateau since preliminary experiments verified that this model performed better in terms of MXinfAP than selecting the model snapshot of the final epoch or the model snapshot with the least loss. The results of our employed approaches are reported in Table 6. The evaluation was conducted using the official test sets of TRECVID SIN 2013 with 112677 images (2nd column), TRECVID SIN 2014 with 107806 images (3rd column) and TRECVID SIN 2015 with 113046 images (4th column). We compare our approaches as well as combinations of them, resulting from max or average pooling the concept probabilities of individual models. We observe that averaging the concept probabilities from our two selected approaches achieves performance that is competitive to that of the method we used before in ReTV; but processing time measurements showed that the inference times of the new models is at least 200% faster than using the older model. Also, the older implementation has a fixed overhead of approximately 5 seconds, since the procedure included calling an external executable and having to load the models into memory for each session, which made it inefficient for repeated use in our service.

Table 6: Evaluation of our approaches on training a model for the SIN concept pool.

Model	*SIN 2013* MXinfAP%	*SIN 2014* MXinfAP%	*SIN 2015* MXinfAP%
B1_1KFPS	27.27	23.27	20.41
B3_1KFPS	27.94	24.79	21.19
avg of B1_1KFPS+B3_1KFPS	**30.04**	**26.08**	**22.94**
max of B1_1KFPS+B3_1KFPS	28.63	25.07	21.52

Pruning Techniques for Neural Networks Results

The proposed fractional step discriminant pruning (FSDP) approach was first evaluated for the task of concept-based annotation using two image and one speech datasets as described in the following: i) CIFAR-10 [25]: It consists of 50000 training and 10000 testing color images of 32×32 resolution, belonging to one of 10 different classes. ii) ImageNet32 [6]: This dataset consists of the annotated images of ILSVRC-2012 resized to 32×32 resolution. It contains 1331167 images in total belonging to one of 1000 classes. It is divided to a training and testing partition of 1281167 and 50000 images, respectively. iii) GSC [51]: This is version 0.01 of the Google speech commands dataset. It consists of 64727 utterances and 12 categories representing short commands such as "No", "Up" and "Go". A training, validation and testing partition is provided consisting of 51094, 6798 and 6835 utterances, respectively.

In the CIFAR-10 dataset the proposed FSDP is compared against several top-performing approaches, namely, MIL [8], PFEC [27], CP [18], SFP [16], ASFP [15] and FPGM [17]. Three

Table 7: Accuracy rates on CIFAR-10 along different ResNet architectures and pruning rate 40%.

	ResNet-20	*ResNet-56*	*ResNet-110*
no pruning	92.2%	93.59%	93.68%
MIL [8]	91.43%	–	93.44%
PFEC [27]	–	91.31%	92.94%
CP [18]	–	90.90%	–
SFP [16]	90.83%	92.26%	93.38%
ASFP [15]	–	92.44%	93.20%
FPGM [17]	91.99%	92.89%	93.85%
FSDP ($\hat{\theta}_f = 10\%$)	92.02%	**93.13%**	93.91%
FSDP ($\hat{\theta}_f = 40\%$)	**92.09%**	93.1%	**93.99%**

popular ResNet architectures (ResNet-20, -56, -110) [14] along different pruning rates θ are used in the evaluation, following the experimental setup in [16, 15, 17]. Specifically, each image is normalized to zero mean and unit variance, and data augmentation is applied during training, i.e., 32×32 random cropping and horizontal flipping with 50% probability. The networks are trained using minibatch stochastic gradient descent (SGD) with Nesterov momentum of 0.9, batch size of 128 and weight decay of 0.0005. The total number of epochs is set to $\epsilon = 200$ and the learning rate starts at 0.01 and is divided by 5 at epochs 60, 120 and 160, as in [16, 15, 17].

In the ImageNet32 dataset experiment, FSDP is compared against SFP [16] and FPGM [17] using ResNet-56, pruning rates $\theta = 20\%, 50\%$ and number of epochs $\epsilon = 40$. Following [6], each image is normalized to zero mean and the minibatch SGD is used, where momentum, batch size and weight decay are set to 0.9, 128 and 0.0005, respectively. The learning rate starts at 0.01 and is multiplied with 0.1 every 10 epochs.

In the GSC dataset, FSDP, SFP [16] and FPGM [17] are evaluated for the task of speech command classification using ResNet-56 and pruning rates $\theta = 20\%, 50\%$. The training and validation sets of GSC are used for training, while the testing set is used for the performance evaluation. Log mel-spectrograms (LMSs) are used to represent the speech commands in GSC. More specifically, the one-second long speech recordings are re-sampled to 16 KHz, the STFT with Hamming window of size 1024 and hop length 512 is applied, and subsequently, 32 mel filterbanks and the logarithmic operator are used to map the power spectra to the log-mel space, and retrieve a 32×32 LMS for each recording. Moreover, following [44], the training dataset is augmented using time-stretching, pitch-shifting and mixing with background noise. The network is trained using minibatch SGD, number of epochs $\epsilon = 70$, and with momentum, weight decay and batch size set to 0.9, 0.0005 and 96, respectively. The initial learning rate starts at 0.01 and is divided by 10 at epoch 50.

In the various experiments, the following input parameters for FSDP (Algorithm 1) are used: i) the asymptotic schedule parameter δ for the estimation of α, β and γ in (10) is set to $\delta = \frac{1}{8}$ as in [15], ii) in order to gain insight into the effect of the target pruning rate associated with the discriminant criterion $\hat{\theta}_f$ (13), we test different values of it, i.e., 10%, 40% on CIFAR-10 (see Tables 7, 8), and in all other experiments this is set to 10%, as explained in Section 4.2.3. Moreover, the between-class scatter matrix $\mathbf{S}$ (6) in the discriminant criterion (5) is computed using the whole training set for CIFAR-10, 20% of the training set for ImageNet32 and the validation set (which is part of the overall training set) for GSC.

The evaluation results in terms of accuracy rates along the different pruning methods on

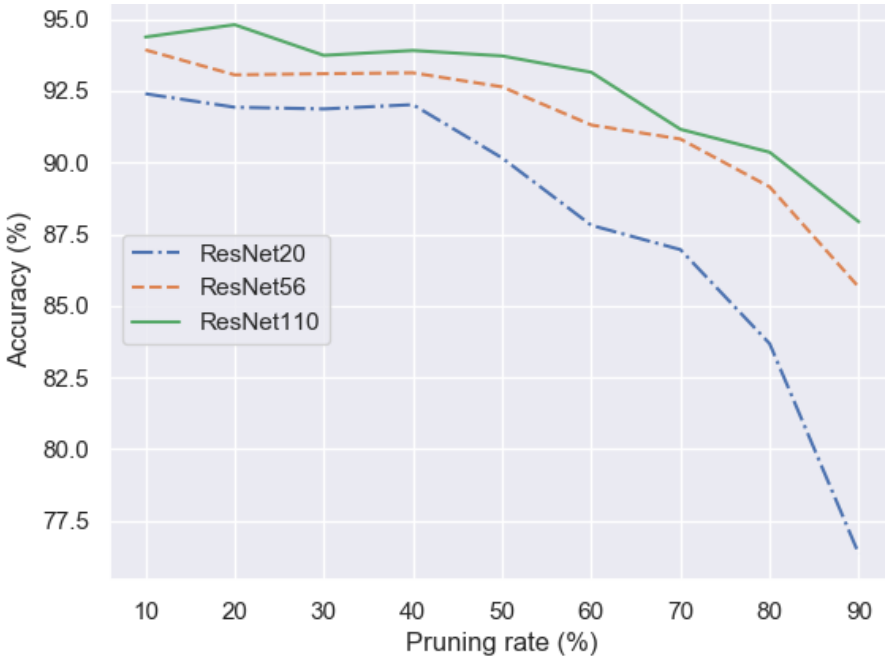

Figure 3: Accuracy rates of FSDP with $\hat{\theta}_f = 10\%$ on CIFAR-10 along different ResNet architectures and pruning rates.

Table 8: Accuracy rates on CIFAR-10 along different ResNet architectures and pruning rate 50%.

	ResNet-20	*ResNet-56*	*ResNet-110*
FPGM [17]	89.73%	91.79%	92.51%
FSDP ($\hat{\theta}_f = 10\%$)	**90.16%**	**92.64%**	**93.72%**

Table 9: Accuracy rates in ImageNet32 and GSC with ResNet-56 and pruning rate 20%.

	ImageNet32	*GSC*
no pruning	40.79%	97.47%
SFP [16]	29.92%	94.57%
FPGM [17]	37.23%	95.64%
FSDP ($\hat{\theta}_f = 10\%$)	**38.3%**	**96.22%**

Table 10: Accuracy rates of FSDP and FPGM in ImageNet32 and GSC with ResNet-56 and pruning rate 50%.

	ImageNet32	*GSC*
FPGM [17]	32.32%	92.89%
FSDP ($\hat{\theta}_f = 10\%$)	**33.23%**	**94.66%**

CIFAR-10, three ResNet architectures (ResNet-20, ResNet-56, ResNet-110) and pruning rate 40% are shown in Table 7, while, Table 8 depicts the accuracy rates of the proposed FSDP and FPGM under the same settings and pruning rate 50%. In Figure 3, we vary the pruning rate from 10% to 90% in order to study its effect in the performance of the proposed method along different network depths. Finally, the evaluation results on ImageNet32 and GSC using ResNet-56 and pruning rates 20% and 50% are shown in Tables 9 and 10. From the obtained results we conclude the following:

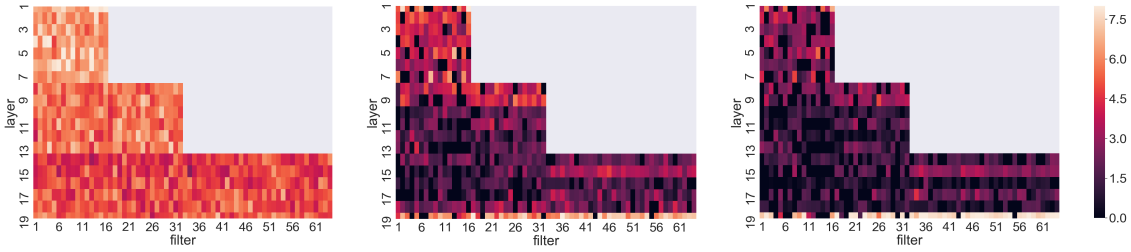

Figure 4: Heatmaps depicting discriminant scores of ResNet-20 filters computed using the discriminant criterion (5) during the application of FSDP on CIFAR-10 with $\theta = \hat{\theta}_f = 20\%$; the heatmaps (left to right) correspond to epochs 1, 40 and 200.

i) The proposed FSDP achieves the best performance in all experiments. For instance, with 50% pruning an accuracy improvement of more than 1% of FSDP over FPGM is observed for ResNet-110 in the CIFAR-10 (Table 8) and for ResNet-56 in the GSC dataset (Table 10). Similarly, in ImageNet32 (Table 9), the proposed FSDP with $\hat{\theta}_f = 10\%$ outperforms the other two methods. This is attributed to the stability of the fractional pruning procedure and the ability of the proposed discriminant criterion to identify the filters with negligible discriminant information. We also see that that the application of FSDP to ResNet-110 in the CIFAR-10 experiments (Tables 7 and 8) yields an increase in performance over the baseline model without pruning. Therefore, we may conclude that the use of FSDP to reduce the capacity of large networks has a regularization effect. This is not observed in the experiments with the smaller networks (ResNet-20, -56) where as expected pruning reduces slightly the performance. Another interesting observation is that SFP appears to have a more than 10% performance drop in ImageNet32 (Table 9), which is due to the much more challenging problem, where a percentage of the pruned filters selected using the l_2 norm criterion still carry important discriminant information.

ii) As shown in Fig. 3, FSDP provides a quite high robustness for pruning rates less than 40%. We also observe that ResNet-110 exhibits the more stable behavior, with even an increase in performance for pruning rates less than 25%, only a small performance drop for pruning rates 40% and 50%, and accuracy of more than 90% even with 80% pruning rate. Finally, we see that the performance of ResNet-20 degrades rapidly with pruning rates higher than 40%, which indicates that network depth is an important parameter concerning the robustness of a network against pruning.

iii) Concerning training times, an overhead of several seconds to a few minutes at epoch level (depending on the dataset) is observed when FSDP is used. For instance, on CIFAR-10 and ResNet-20 this time is increased from 17 to 42 seconds, while for ResNet-110 an overhead of 1.5 minutes, i.e. going up from 1.7 to 3.2 minutes, is observed. However, concerning that the training is performed off-line, its duration is less than a day in all experiments, and that the computation of the discriminant criterion (5) can be accelerated significantly using a more powerful GPU, this time overhead is considered insignificant.

In order to gain further insight into the proposed framework, Fig. 4 illustrates the discriminant scores of each filter along different epochs during the training of ResNet-20 with FSDP on CIFAR-10 and pruning rates $\theta = \hat{\theta}_f = 20\%$ (i.e. only the discriminant criterion (5) is used for filter selection and pruning in this experiment). The heatmaps on the left, middle and right correspond to the state of the network at epochs 1, 40 and 200, while, the x-, y-axis at each heatmap represent the filter and convolutional layer number, respectively. A cool-to-

warm color spectrum is used to represent the filter discriminant score at logarithmic-scale, i.e., darker squares represent less important filters while lighter ones correspond to filters with a high discriminant score. We can observe the following:

i) Filters at layers closer to the network input seem to attain a relatively higher discriminant score. This phenomenon seems to be stronger during the first stages of the training procedure and becoming less emphatic as the network converges to its steady-state condition. We also observe a possible correlation between filter's discriminant score and layer's width, i.e., the discriminant power of a filter seems to decrease with the number of filters in the layer it belongs to. For instance, we see that filters of layers 1 to 7 attain in general a higher discriminant score in comparison to the filters of layers 8 to 13. These conclusions are in agreement with similar ones in other literature works, e.g. [33], where it is stated that "global importance seems to decrease with depth".

ii) The last convolutional layer (layer 19) is a clear exception to the above observation, where we see that after a certain number of epochs the majority of its filters attain a large discriminant score. Similarly, another exception are the filters of the second convolutional layers in residual blocks, which also attain a relatively high discriminant score. For instance, this can be seen more clearly at the final network (right heatmap in Fig. 4) for the filters in layers 11, 13, 15 and 17. This conclusion is again in agreement with similar ones in the literature (e.g. see Section 4.3 in [27]).

iii) At epoch 40, most discriminant information has been already concentrated in a rather small portion of the filters. From this, we can conclude that there is a quite high redundancy in the network for the specified problem, and that the proposed approach can effectively discover a more compact network structure already at the early stages of the training.

Next, we proceed to the evaluation of the LSTM pruning variant of the proposed method for the task of concept detection using the large-scale YouTube8M (YT8M) dataset [26]. This is the largest publicly available multi-label video dataset consisting of 6134598 videos annotated with one or more labels from 3862 classes. It is already divided to a training and validation partition, consisting of 3888919 and 1112356 videos, respectively. Moreover, visual and audio feature vectors in $\mathbb{R}^{1024}$ and $\mathbb{R}^{128}$, respectively, are already provided at frame-level granularity.

Table 11: Evaluation results in the YouTube8M dataset.

	GAP@20
SDCNN [11]	82.2%
Proposed (no pruning)	84.3%
Proposed ($\theta = 10\%$)	84.3%
Proposed ($\theta = 20\%$)	84.1%
Proposed ($\theta = 30\%$)	84%
Proposed ($\theta = 40\%$)	83.5%
Proposed ($\theta = 50\%$)	83.5%

For the evaluation in this task we use an RNN architecture consisting of a bidirectional LSTM (BLSTM) [13], an LSTM and a sigmoid (SG) layer. The forward and backward layers of the BLSTM contain 512 cells each, while 1024 cells are used for the LSTM layer. The RNN is trained for 10 epochs using minibatch stochastic gradient descent (SGD), batch size of 256, cross entropy (CE) loss, an exponential learning rate schedule with initial learning rate of 0.0002 and learning rate decay of 0.95 every epoch. We tested for different pruning rates θ,

specifically for 10%, 20%, 30%, 40% and 50%. In all the experiments pruning was applied every 200 training iterations as well as after the last iteration of each epoch. The YT8M videos are represented as vector sequences in $\mathbb{R}^{1152}$ resulted by the concatenation of the respective frame-level visual and audio feature vectors. Finally, the global average precision at 20 (GAP@20) is used as the performance evaluation metric, which is the primary evaluation metric of the YT8M challenge [26].

The evaluation results are shown in Table 11. From the obtained results we observe our new RNN-based approach without pruning surpasses our previous method (SDCNN) by more than 2% GAP. This is because, in contrast to SDCNN that uses only video-level feature vectors derived with mean-pooling, our new approach can effectively exploit the discriminant temporal information in the video sequences. We also see that the the proposed pruning approach is very stable, causing negligible performance drop for small pruning rates ($\theta < 30\%$), and a rather small performance degradation for larger pruning rates.

"Sandmännchen and Friends" Character Identification Results

Table 12: "Sandmännchen and Friends" character frame-level identification results per class.

Character	*Precision*	*Recall*	*F1-score*	*Accuracy*
HerrFuchs&FrauElster	0.530	0.930	0.680	0.929
Jan&Henry	0.800	0.600	0.680	0.595
Kalli	0.720	0.840	0.770	0.838
KleineKonig	0.840	0.890	0.860	0.885
KleineRabeSocke	1.000	0.570	0.730	0.573
Luzi&Moffels	0.830	0.580	0.680	0.580
MeineMchmusedecke	0.460	0.510	0.490	0.514
Pittiplatsch&Schnatterinchen&Moppi	0.750	0.560	0.640	0.558
Plumps	0.700	0.910	0.790	0.912
Pondorondo	0.720	0.990	0.830	0.987
Rita&Krokodil	0.550	0.980	0.710	0.980

Table 13: "Sandmännchen and Friends" character video-level identification results per class.

Character	*Precision*	*Recall*	*F1-score*	*Accuracy*
HerrFuchs&FrauElster	1.000	1.000	1.000	1.000
Jan&Henry	1.000	1.000	1.000	1.000
Kalli	1.000	1.000	1.000	1.000
KleineKonig	1.000	1.000	1.000	1.000
KleineRabeSocke	1.000	1.000	1.000	1.000
Luzi&Moffels	1.000	1.000	1.000	1.000
MeineMchmusedecke	1.000	1.000	1.000	1.000
Pittiplatsch&Schnatterinchen&Moppi	1.000	1.000	1.000	1.000
Plumps	1.000	1.000	1.000	1.000
Pondorondo	1.000	1.000	1.000	1.000
Rita&Krokodil	1.000	1.000	1.000	1.000

In Tables 12 and 13 we report the evaluation results for the "Sandmännchen and Friends" character identification frame-level and video-level predictions, respectively. We observer that after employing the additional domain rules to infer video-level predictions we achieve a perfect score. This observation is also backed from the frame-level (Table 15) and video-level (Table 13) confusion matrices.

Table 14: Confusion matrix for "Sandmännchen and Friends" characters frame-level identification.

	HerrFuchs&FrauElster	*Jan&Henry*	*Kalli*	*KleineKonig*	*KleineRabeSocke*	*Luzi&Moffels*	*MeineSchmusedecke*	*Pittiplatsch&...*	*Plumps*	*Pondorondo*	*Rita&Krokodil*
HerrFuchs&FrauElster	92.8	0.0	0.0	0.0	0.0	0.0	0.0	7.1	0.0	0.0	0.0
Jan&Henry	0.58	59.5	9.25	1.7	0.0	0.5	6.9	4.0	2.3	3.4	11.5
Kalli	1.1	0.3	83.7	1.9	0.0	1.5	5.0	0.3	1.1	0.3	4.2
KleineKonig	0.0	0.0	0.0	88.5	0.0	0.0	4.1	0.0	0.0	1.0	6.2
KleineRabeSocke	0.0	0.0	5.1	1.7	57.2	0.0	14.5	0.0	0.8	0.8	19.6
Luzi&Moffels	3.11	2.33	11.28	1.9	0.0	57.9	5.8	2.7	4.2	6.2	4.3
MeineSchmusedecke	1.8	5.4	12.6	0.0	0.0	14.4	51.3	6.3	1.8	2.7	3.6
Pittiplatsch&...	5.8	7.7	12.3	0.6	0.0	6.4	1.3	55.8	6.4	1.3	1.9
Plumps	0.00	1.2	0.0	0.0	0.0	0.0	0.0	6.2	91.2	1.2	0.0
Pondorondo	0.0	0.0	0.0	0.0	0.0	0.0	1.2	0.0	0.0	98.7	0.0
Rita&Krokodil	0.0	0.0	0.0	0.0	0.0	0.0	2.0	0.0	0.0	0.0	97.9

Table 15: Confusion matrix for "Sandmännchen and Friends" characters video-level identification.

	HerrFuchs&FrauElster	*Jan&Henry*	*Kalli*	*KleineKonig*	*kleine rabe socke*	*Luzi&Moffels*	*MeineSchmusedecke*	*Pittiplatsch&...*	*Plumps*	*Pondorondo*	*Rita&Krokodil*
HerrFuchs&FrauElster	100	0	0	0	0	0	0	0	0	0	0
Jan&Henry	0	100	0	0	0	0	0	0	0	0	0
Kalli	0	0	100	0	0	0	0	0	0	0	0
KleineKonig	0	0	0	100	0	0	0	0	0	0	0
KleineRabeSocke	0	0	0	0	100	0	0	0	0	0	0
Luzi&Moffels	0	0	0	0	0	100	0	0	0	0	0
MeineSchmusedecke	0	0	0	0	0	0	100	0	0	0	0
Pittiplatsch&...	0	0	0	0	0	0	0	100	0	0	0
Plumps	0	0	0	0	0	0	0	0	100	0	0
Pondorondo	0	0	0	0	0	0	0	0	0	100	0
Rita&Krokodil	0	0	0	0	0	0	0	0	0	0	100

5. Brand Detection

5.1. Updated Problem Statement and State of the Art

For a complete state-of-the-art analysis regarding object detection, the reader is referred to D1.1 (Section 5.2) of ReTV. In this deliverable we discuss our efforts for months M21 to M30, which focused on a) extending the pool of detectable brands, b) optimizing the performance of the brand detection module of the VA service, and c) employing new models for a car brand detection use-case.

5.2. Extended Brand Pools for Video Annotation

After discussion with ReTV partners we decided to expand the set of detected brands, in order to support the use-case scenarios. The "Berlin" brands set was requested by RBB. It contains 95 logos of Berlin brands like AdmiralSP, Berliner Kindl, BVG, Eisbaeren, Mustafas and others. To be able to train a model on the "Berlin" brands set we used the methodology described in Section 5.2 of D1.2 which employs a Web-crawler to fetch 20 images from image search engines and a data augmentation scheme to form a dataset of 1000 images for each brand logo, of which 900 images are used for training the object detector and the rest for the validation of the model during the training procedure.

5.3. Brand Detection Performance Optimization

In the context of our efforts on optimizing the performance of the brand detection module we decided to use a new, merged model for all logo detection tasks. Using a single model we jointly detect the logos of the five brand pools that were deployed during the course of the project, specifically: a) the baseline ReTV 225 brands set, collected by CERTH and discussed in D1.1, b) the 53 football teams logos set, requested and collected by RBB, discussed in D1.2, c) the 95 Berlin brands logos set, requested by RBB, collected by CERTH and discussed here in D1.3, d) the 26 Swiss brands logos set, requested by GENISTAT, collected by CERTH and discussed in D1.2, e) the 18 Swiss channel logos set, requested by GENISTAT, collected by CERTH using a dataset provided by GENISTAT and discussed in D1.2. In total the new model can detect over 400 different logos of brands, football teams and channels.

To train the new model, we choose to once again use the RetinaNet [29] framework since this represents a good compromise between time-efficiency and effectiveness, as discussed in Section 5.1 of D1.2. We also used the exact same training and validation datasets utilized for the training of the respective older individual models.

This way we managed to improve speed, since a single inference from one model can provide the detections for objects of all employed pools. Taking into consideration that the size that a DCNN object detection model occupies in the GPU memory depends on the selected architecture and is not directly related to the number of objects it can detect and classify, it is obvious that we also managed to significantly reduce the memory footprint of the brand detection module.

5.4. Car Brand Detection

In discussions between CERTH and MODUL, the need for signals to support a car brand detection scenario came up. Specifically, the requirement was to be able to detect cars in a video stream and the cars brand. We designed a method where first we employ an object

detector to detect the car in the frames of a video. Then, after cropping the input frame to the bounding box of a detected car, we feed the car image region to a different DCNN to infer the car brand.

Initially, we setup a car brand detection framework (CBDF1) where for the car detection we used the RetinaNet architecture [29], pre-trained on the Microsoft COCO dataset [30], available on the internet. This dataset contains the "car" object class and therefore suits our need to detect cars in video frames. For the car brand classification we trained three DCNN models with different architectures (namely, a DenseNet201 [21] model, an InceptionV3 [47] model, and an Xception [5] model) on the StanfordCars [24] training set which contains 196 models. We also implemented an ensemble technique employing all three different DCNN model architecture by max pooling their inferred probabilities. The ensemble scheme achieved a 89.51% accuracy on StanfordCars test set, yet it was deemed that certain recent, as well as some older but iconic car brands, were not included in the detectable car models set. Therefore, in a next attempt (CBDF2), we merged the StanfordCars models with a selection of models from the VMMRdb dataset [48], detecting in total a set of 559 models of 54 different brands. Our criteria for selecting the subset of VMMRdb car brands were: a) the car model must have at least 100 training images (due to the notorious data-hungry nature of DCNNs), b) the release date must be after 2000, excluding a set of certain specific models (such as the Volkswagen Beetle) which are older yet iconic, and c) a visual inspection of training images must verify the diversity of training images. We trained the same three DCNN model architectures. This ensemble scheme attained an accuracy of 90.52% on the StanfordCars test set.

In a later month, MODUL also manually annotated 2 hours of TV programming to test the car brand detection on ReTV-relevant data. The automatic results were then compared with the manual 'ground truth' annotation in order to fine tune the configuration of the service. Noting that in manual annotation, only cars which were clearly visible for a few seconds in the program were actually recorded, we decided on a more concise format where a minimum time duration was needed between the appearance time of car and until it is no longer visible, for it to be included in the detection results. This removed many false positives (as cars appearing for too short duration were also more difficult to detect correctly, and unnecessary as the viewer would also be much less likely to note the brand correctly). We chose a minimum time of one second as we found at two seconds there was a further improvement in precision but at a greater cost to recall. Therefore, for our final setup (CBDF3), and in accordance with the collected feedback, we decided to proceed to the following modifications to our car brand detection system:

- Drop the car model inference - instead only infer the respective car brand. After this change, our car brand detection framework can detect 59 brands.
- Balance the number of training images for each class by adding more samples for some "neglected" classes (e.g. Tesla) in order to further improve the classification accuracy.
- Trained new models on the adjusted training set employing SoA architectures, namely the EfficientNet-B1, EfficientNet-B3 and EfficientNet-B5 [49].
- Change the object detection framework from RetinaNet to YOLO v3 [40], again using a pre-trained model on the Microsoft COCO dataset available on the internet. We used a faster detection stage resulting in real-time overall analysis speed. The YOLO v3 detector is approximately 3 times faster with a slightly better accuracy (51.5% mAP in the COCO evaluation set versus a 50.3 mAP% of the RetinaNet on the same dataset), resulting in a processing time which is less than the input video's duration (faster than

real-time).

- Implement a simple "tracking" scheme: each detected car gets a unique ID and if the intersection-over-union measure of the bounding box of a newly detected car to a bounding box of a car detected in the previous frame is over a pre-defined threshold, it gets the same ID. The tracking scheme not only improves the presentation of the results, but also increases the car brand classification, since we perform max pooling of the brand predictions for each detected car under the same ID.
- Fixed the optimal parameters of the car brand detection (namely, rate of frame sampling from the video, car detection threshold, car classification threshold, minimum area and time needed for a car detection to be considered valid) based on the analysis of experimental results.

The evaluation results for all three implemented car brand detection frameworks (CBDF1, CBDF2 and CBDF3) are reported in Section 5.6.

5.5. Implementation Details and Use

We used the Keras[10] open-source neural-network library version 2.3.1, with the TensorFlow backend[11] version 1.14 for designing and training our DCNN models. Brand detection experiments were conducted on an Intel i7 3770K server with 32GB of RAM, running Ubuntu 18, equipped with an Nvidia GeForce GPU (GTX 1080 Ti). Car brand detection experiments were conducted on an Intel i5 3570K server with 32GB of RAM, running Windows 10, equipped with an Nvidia GeForce GPU (Tesla K40).

5.6. Results

Table 16: Evaluation of the various versions of brand detection module on literature datasets that contain logos supported by the VA service.

Dataset	Max detection score reported in the literature	Detection score of the ReTV method in D1.1	Detection score of the ReTV method in D1.2	Detection score of the current ReTV method
LogosInTheWild [50]	84.2% mAP at closed set, 46.4% at open set [50]	79.8% mAP at closed set	92.2% mAP at closed set	**92.6%** mAP at closed set
WebLogo-2M [45]	34.37% mAP [45]	N/A (only test set is provided and is used for training in our case)		
TopLogo10 [46]	41.8% mAP [46]	53.3% mAP	83.3% mAP	**86.3%** mAP
Logos-32plus [2]	94.5% F-score, 95.8% ac. [2]	94.13% F-score, 92% acc.	**97.25%** F-score, **98.83%** acc.	95.29% F-score, 97.32% acc.
FlickrLogos-47 [12]	48.1% mAP [39]	27.84% mAP	58.57% mAP	**59.64**% mAP
FlickrLogos-32 [43]	90.3% F-score [42]	94.13% F-score	**98.26%** F-score	**98.26%** F-score
FlickrLogos-27 [23]	53% acc. [23]	81.5% acc.	**91.5%** acc.	**91.5%** acc.
BelgaLogos [22]	34.11% mAP [22]	23.11% mAP	**41.11%** mAP	39.21% mAP

In Table 16 we present the evaluation of all the adopted object detection approaches throughout the course of ReTV. We compare against the datasets which were incorporated in the ReTV baseline brands set and have a separate test set. Reported in the second column of this table is the score of the methods of the literature that achieve the best score to our knowledge

[10] https://keras.io/
[11] https://www.tensorflow.org/
[12] http://www.multimedia-computing.de/flickrlogos/

on the respective datasets. In the third, fourth and fifth columns we report the evaluation results for our previous object detection methods from D1.1, D1.2 and the current deliverable, respectively. We conducted the evaluation by calculating each time the measure that the respective literature method used (F-score as the harmonic mean of the Precision and Recall measures, mean average precision - denoted as "mAP" - and accuracy - denoted as "acc."). We observe that our implementations have the best score for all considered datasets. The final method is not the best in terms of performance amongst our older implemented methods. However, this marginal drop in accuracy is not significant when taking into consideration the great gains in efficiency.

Finally, in Table 17 we report the evaluation results on the StanfordCars test set, for the various implement frameworks (CBDF1, CBDF2 and CBDF3). Note, that for CBDF1 and CBDF2 approaches, we include the accuracy on inferring each cars model. In CBDF3, this not applicable since this framework only predicts a car brand and not a specific model. We also compare our results to the reported results of [24], which achieves the best accuracy on the StanfordCars dataset, to our knowledge. However, for all approaches, we also evaluate the accuracy when dealing only with car brands (column 5). We observe that for all approaches the ensemble technique scores a better accuracy. Additionally, the model prediction accuracy was increased from 89.95% in CBDF1 to 90.52% in CBDF2 even though the pool of detectable models was greatly expanded with models that are not included in the test set.

Table 17: Car brand detection results.

Method	*Number of classes*	*DNN architecture used*	*Car model detection accuracy (%)*	*Car brand detection accuracy (%)*	*Mean classification time per image (milliseconds)*
CBDF1	196 models	DenseNet201	86.75	91.52	40.84
		InceptionV3	84.72%	85.55	25.52
		Xception	86.65	86.42	31.44
		All	89.95	93.92	161.25
CBDF2	559 models	DenseNet201	85.42	91.86	39.78
		InceptionV3	88.09	90.26	25.45
		Xception	83.55	90.05	31.57
		All	90.52	86.52	86.25
CBDF3	41 brands	EfficientNetB1	N/A	92.16	19.44
		EfficientNetB3	N/A	90.56	24.32
		EfficientNetB5	N/A	91.82	42.56
		All	N/A	**94.19**	91.27
[24]	196	N/A	**94.50**	N/A	Not reported

It is worth noting that in [24] the 3D geometry of cars is estimated from images via 3D reconstruction. Even though time measurements of this process are not reported in this paper, taking into consideration the description of their method, we can safely assume that it is significantly slower compared to our method. This trait of efficiency is particularly important in our case, since our car brand detection scenario deals with long videos rather than a collection of a few still images.

We also made a small internal evaluation with two one-hour TV programs. We chose two episodes of a TV program about cars on German channel VOX, mainly because it would be certain in this program to have multiple appearances of car brands with a significant presence on screen. We compared the initial service with the service configured to only report detected

brands where the car is on screen for at least one second. The initial service had a precision of 0.08 (there were 201 detections of a car brand, of which 17 were correct annotations and still 10 occurrences were missed) with a recall of 0.63 (of the 27 manually annotated occurrences, 17 were correct). Comparing this with the service configured with the minimum time, the precision was 0.2 at 1 sec minimum and actually peaked at 0.29 at 2 seconds, however the recall at 1 second was 0.52 which dropped for longer minimum times to 0.33. As a result, we preferred the one second configuration which led to an optimal f1 score (harmonic mean between precision and recall) of 0.29, and which was a significant improvement on the service with no minimum time whose f1 score had been 0.14.

6. Updated Video Analysis Component, Workflow and API

6.1. Video Analysis Component Updated Functionalities and Outputs

The video fragmentation techniques discussed in Section 4.1, the concept-based video abstractions discussed in Section 4.2, as well as the brand detection method discussed in Section 5, have all been incorporated into the Video Analysis component. This component was deployed as a REST service hosted by CERTH servers prior to M21, and is described in detail in D1.2 (Section 6.1). Here, we mention the updates conducted from months M21 to M30.

Towards a more production-ready stage of CERTH services, we implemented two necessary additions. Firstly, we established a domain name (`retv.iti.gr`) for the CERTH server that hosts the Video Analysis service of WP1 (as well as the Video Summarization service of WP3). We allowed to all ReTV applications that call the CERTH services' endpoints a transition period until the end of October, 2019. The establishment of a domain name allows CERTH to a) hide the underlying IP of the server that hosts the services and b) point the domain name to a different IP of a backup server that hosts a clone of the CERTH services, allowing for a quick recovery in case of hardware failure and without the need for partners to adjust their calls.

Secondly, we established an authentication process in WP1 (and WP3) services. Authentication keys were provided to all partners. The provided key is a hash code, i.e., a 32-character length string and should be included in a "user_key" field of the JSON-structured body of the HTTP call for all endpoints of CERTH services. Again, there was a transition period until the end of October of 2019, where all applications of ReTV could optionally use this authentication process, until it was finally enforced.

CERTH has developed a text to video matching method which deals with the transformation of the typical image/video and text representations into a new common embedding space, in which the similarity between image and text can be directly measurable. The development of the text to video module is part of the Content Adaptation and Re-Purposing task (T3.3) of WP3. Our method is presented in detail in D3.3. Yet, since for practical reasons and as foreseen the final architecture of the ReTV services the extraction of text to video embeddings is conducted by the Video Analysis service of WP1, we describe here the available endpoints.

We have implemented two endpoints for text to video embeddings extraction from videos and one for text to video embeddings extraction from text. Regarding the video embeddings' extraction this can be done by either calling the `va` endpoint (`retv.iti.gr:8090/va`) setting the "text2video" parameter to 1 in the JSON-structured body of the HTTP call, or by calling the `t2v` endpoint (`retv.iti.gr:8090/t2v`). In the latter case the Video Analysis service only performs temporal segmentation to shots and generates the text to video embeddings for each shot, leading to a 200% speedup of the process. A user can use the `t2v` endpoint when all the other analysis results of the VA service (i.e. concept-based annotation, object detection and content-type classification results) are not needed.

Concerning the text to video embeddings extraction from text, the `t2vs` (`retv.iti.gr:8090/t2vs`) endpoint of the Video Analysis service can be used. This accepts a text string and produces one text to video signature vector for each sentence (ending with a full stop character). If a longer text, made of multiple sentences, is submitted, the service will produce a text to video signature for each separate sentence. Text to video embeddings of both videos and text, are 2048-element vectors and are saved using the Apache Parquet output format in the session and are available for download for 48 hours - the time that a session results folder

remains in CERTH's repository before it is deleted. The permanent storage and comparison of embeddings is conducted by GENISTAT, as discussed in Section 2.3 of D3.3.

Additionally, we constructed a new Image Collection Analysis (ICA) endpoint of the Video Analysis service, available at `retv.iti.gr:8090/ica`. It receives a compressed collection (zip file) of images and performs concept detection and brand detection on each image. This was mainly developed for debugging purposes, specifically, a) for CERTH to quickly verify the function of the concept and object detection modules and b) for ReTV partners to test the results of the aforementioned modules.

Finally, the car brand detection was initially available at `retv.iti.gr:8090/cd`; the calls to it go to the same queue as those for the main VA service (namely the `va`, `t2v`, `t2vs` and `ica` endpoints). We later moved the car brand detection to a different server due to hardware limitations and it is now available at `160.40.50.245/cd`. As the queue of calls in `160.40.50.245/cd` is utilized only for the calls to the `cd` endpoint, the calls are immediately processed, since no other endpoints are hosted by this newly used server.

6.2. Component Updated API and Usage Instructions

The API and usage instructions for the developed REST services is described in detail in D1.2 (Section 6.2 and 6.3). Here we present the updated information for certain new endpoints and parameters.

New parameters for the va endpoint

For the

```
retv.iti.gr:8090/saf
```

endpoint the list of optional JSON structured arguments in the POST call body was expanded to adopt the newly introduced functionality and now includes:

- "brand_detection" (accepts an integer in the range [0,1] with default value=1): Set to 0 to disable brand detection for the current session, resulting in faster execution times.
- "saf" (accepts an integer in the range [0,1] with default value=0): Set to 1 to enable concept detection on the "Sandmännchen and Friends" related signals extraction (see Section 4.2.1), besides the normal operation of the `va` endpoint. If this argument is set to 1, in the video analysis results JSON there will a "SaF_info" element with the "character_ind", "character_label", "character_prob", "intro_shots", "main_shots", "outro_shots" items providing all extracted information regarding a "Sandmännchen and Friends" video.
- "lbvs" (accepts an integer in the range [0,1] with default value=0): Besides the normal operation of the `va` endpoint, additionally extract the text to video feature vectors for each shot of the video.
- "lbvs" (accepts an integer in the range [0,1] with default value=0): Besides the normal operation of the `va` endpoint, additionally extract learning-based video summarization features to be later used from the Video Summarizaton service of WP3.

Text to video embeddings extraction endpoints

As discussed in Section 6.1 we implemented several endpoints to support the signature extraction for the Text to video method of WP3. Besides calling the older `va` endpoint setting the "text2video" parameters to 1, we can also use the additional two endpoints:

1. `retv.iti.gr:8090/t2v`

   The obligatory JSON structured arguments that must be included in the POST call body are:

   - "user_key": The key is a 32 character hash string provided by CERTH for the authorization of the call.
   - "video_url": A single video URL from a Web page as input to the service. The video will be downloaded from the Web page and analyzed. This can also be a Google Drive link or any link that directly points to a downloadable video file.

2. `retv.iti.gr:8090/t2vs`

   The obligatory JSON structured arguments that must be included in the POST call body are:

   - "user_key": The key is a 32 character hash string provided by CERTH for the authorization of the call.
   - "text": A single line text. The service will produce one text2video feature vector for each sentence (ending with a full stop) If a longer text, made of multiple sentences, is submitted, the service will produce a text2video feature vector for each separate sentence.

"Sandmännchen and Friends" related signals endpoint

`retv.iti.gr:8090/saf`

This configures the VA service to only extract "Sandmännchen and Friends" related signals. Any other information (e.g, concept-based annotations, object detection results, etc) will not be extracted and will not be available in the results. The syntax of the HTTP POST call for this endpoint is similar to the start call of the `va` endpoint. This endpoint is approximately 170% faster than calling the `va` endpoint with the "saf" parameter set to 1.

Image Collection Analysis endpoint

`retv.iti.gr:8090/ica`

The obligatory JSON structured arguments that must be included in the POST call body are:

- "user_key": The key is a 32 character hash string provided by CERTH for the authorization of the call.
- "zip_url": A URL that points to a zip compressed file of the image collection. This can be a Google Drive link or any link that directly points to a downloadable zip file.

New Status Messages

We briefly remind the reader that the HTTP GET status call to

```
HTTP GET http://retv.iti.gr:8090/<session>/status
```

returns a JSON file (described in Section 6.2 of D1.) If the call is successful, the JSON file contains the fields "status" and "message". From M21 to M30, and after discussions with technical partners of ReTV, we expanded the list of messages in order to make the debugging procedure in case of an error, more straight-forward. The messages are self-explanatory in most cases. Table 18 reports the complete set of possible Status messages.

```
VIDEO ANALYSIS WAITING IN QUEUE
VIDEO COPYING STARTED
VIDEO FIND FAILED
VIDEO COPYING FAILED
VIDEO DOWNLOAD STARTED
VIDEO DOWNLOAD FAILED
VIDEO SEGMENTATION STARTED
VIDEO SEGMENTATION FAILED
VIDEO SEGMENTATION COMPLETED
VIDEO ANALYSIS STARTED
VIDEO ANALYSIS FAILED
VIDEO LBVS FEATURE EXTRACTION STARTED
VIDEO SAF FEATURES AGGREGATING FAILED
VIDEO ANALYSIS FAILED
VIDEO ANALYSIS COMPLETED
TEXT2VIDEO FOR VIDEO WAITING IN QUEUE
TEXT2VIDEO FEATURE EXTRACTION FAILED
TEXT2VIDEO FOR VIDEO COMPLETED
TEXT2VIDEO FOR STRING WAITING IN QUEUE
TEXT2VIDEO FOR STRING FAILED
TEXT2VIDEO FOR STRING COMPLETED
SUMM. WAITING IN QUEUE
SUMM. END2END STARTING ANALYSIS
SUMM. END2END ANALYSIS FAILED
SUMM. FETCHING VIDEO ANALYSIS DATA
SUMM. FAILED. COULD NOT RETRIEVE VIDEO ANALYSIS DATA FROM CERTH
SUMM. FAILED. COULD NOT LOAD VIDEO ANALYSIS DATA FROM GENISTAT
SUMM. SCRIPT CREATION STARTED
SUMM. SCRIPT CREATION FAILED
SUMM. SCRIPT CREATION COMPLETED
SUMM. SCRIPT RENDERING STARTED
SUMM. SCRIPT RENDERING FAILED
SUMM. SCRIPT RENDERING FAILED
SUMM. SCRIPT RENDERING COMPLETED
SUMM. COMPLETED
RENDERING WAITING IN QUEUE
IMAGE COLLECTION ANALYSIS WAITING IN QUEUE
IMAGE COLLECTION DOWNLOAD STARTED
IMAGE COLLECTION DOWNLOAD FAILED
IMAGE COLLECTION ANALYSIS STARTED
IMAGE COLLECTION ANALYSIS FAILED
IMAGE COLLECTION WRITING RESULTS FAILED
IMAGE COLLECTION ANALYSIS COMPLETED
CD WAITING IN QUEUE
CD VIDEO READING AND DETECTION STARTED
CD CLASSIFICATION STARTED
CD CLASSIFICATION FAILED
```

CD AGGREGATING RESULTS
CD WRITING RESULTS
CD WRITING RESULTS FAILED
CD COMPLETED

Table 18: Set of possible status messages that an HTTP GET Status call to the Video Analysis will return

6.3. Component Testing and Software Quality Assessment

The Video Analysis service and its constituent methods have undergone extensive testing, both by their developers and by other ReTV partners that use the service for receiving video analysis results. In terms of benchmark testing of the methods implemented in the service, these results have been reported in the respective results sections of the present deliverable (Sections 4.1.2, 4.2.7, 4.2.7, and 5.6). In terms of unit testing, several hundred such tests were conducted to cover all the different possible call configurations of the updated and extended service (given the number of either mandatory or optional parameters of the REST service API, as detailed in Section 6.2), and also cover edge cases (such as very short or very long videos submitted for analysis; corrupt video files submitted, unsupported formats etc.). In terms of stress testing, as our REST service continues to implement a queuing mechanism, as described in Section 6.3 of D1.2, any number of requests that come in a burst or in general while the service is processing another request will be queued, in a first-in-first-out manner. The queuing mechanism was tested with a few hundred analysis requests (videos) submitted in bursts, and behaved as expected, i.e. all the requests were added to the queue and were processed sequentially.

7. Conclusion and Outlook

In this deliverable, we described the third and final version of the ReTV data ingestion, analysis and annotation components. Data collection is active across multiple vectors and languages, feeding annotated documents into the ReTV metadata repository for subsequent analysis tasks (WP2). In terms of content collection across vectors, and annotation and knowledge graph alignment, we continually evaluated data quality and corrected the data collection pipeline when needed. We added vectors or languages as needed by partners or stakeholders and keyword extraction was improved to disambiguate keywords with entities. Named Entity Linking was improved by full integration with our SKB and extended to better support entities of type Works (e.g. TV series). Interacting with the SKB was improved through the updates to the search capabilities and results display of the SKBBrowser and its extension to allow entity addition/editing (SKBEditor).

The active task of data collection has ended in the ReTV project, however we will continue to monitor and maintain the data collection pipeline. The Web-based TVP Visual Dashboard (branded Topics Compass in our WP5 professional user scenario) will be configured for our new scenarios with the filter in place to allow a focus on data sources classified as youth, culture or sport, respectively. This will be reported subsequently in ReTV WP5. We will continue to improve Recognyze in the next months and even after the end of the current project. Our next release will be focused on improving the results of the NER engine. One of the major improvements will be to use Transformer-based language models to improve these NER results. This will also lead to improvements in the NEL engine, as a significant number of the errors we have observed came directly from the NER engines we have used in the past (e.g., our own engine or Stanford). Since Orbis is a flexible framework and offers an affordable option for building new evaluation use-cases, it can easily be argued that it is in fact a framework designed to help build evaluation infrastructure. The focus on lenses and on understanding the variation in results caused by the different annotation styles, KGs or entity classification schemes differentiates Orbis from other tools for NER/NEL evaluation like GERBIL. An initial evaluation with the lenses related to annotation styles discussed in this paper has shown great potential for further development, therefore we plan to extend the number of lenses for the datasets for a wider number of datasets. We think lenses related to Knowledge Graph versions and information on which KG version is used by every annotator should be included in all published evaluations. Such a requirement, however, might take some time to implement, therefore including KG or entity typing lenses in different evaluation systems might be a good step towards enhancing reproducibility.

Concerning concept-based video abstractions and brand detection, we focused on the performance optimization of all modules of the Video Analysis service, by algorithmic performance optimization efforts and implementation-related optimization actions. As part of the algorithmic performance optimization effort, we worked on compacting DNNs using pruning techniques. In the context of implementation-related performance optimization, we re-designed the bottleneck parts of the service and replaced older, outdated sub-modules using modern and more efficient alternatives. Our efforts to improve the performance of the modules of Video Analysis (VA) service yielded a significant speedup, as documented in Section 4.1.2. The set of brand pools supported by the VA service was also expanded. New functionalities regarding the extraction of signals to support specific use-cases (i.e. structure and character identification for the "Sandmännchen and Friends" scenario and car brand detection) were introduced. Overall the final version of the Video Analysis service described in this deliverable is a mature software component for use in ReTV.

References

[1] K. Apostolidis, E. Apostolidis, and V. Mezaris. A motion-driven approach for fine-grained temporal segmentation of user-generated videos. In *International Conference on Multimedia Modeling*, pages 29–41. Springer, 2018.

[2] S. Bianco, M. Buzzelli, D. Mazzini, and R. Schettini. Deep learning for logo recognition. *Neurocomputing*, 245:23–30, 2017.

[3] A. M. P. Brașoveanu, G. Rizzo, P. Kuntschick, A. Weichselbraun, and L. J. Nixon. Framing named entity linking error types. In N. Calzolari, K. Choukri, C. Cieri, T. Declerck, S. Goggi, K. Hasida, H. Isahara, B. Maegaard, J. Mariani, H. Mazo, A. Moreno, J. Odijk, S. Piperidis, and T. Tokunaga, editors, *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, pages 266–271, Paris, France, may 2018. European Language Resources Association (ELRA).

[4] Y. Cheng, D. Wang, P. Zhou, and T. Zhang. Model compression and acceleration for deep neural networks: The principles, progress, and challenges. *IEEE Signal Processing Magazine*, 35(1):126–136, 2018.

[5] F. Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017.

[6] P. Chrabaszcz, I. Loshchilov, and F. Hutter. A downsampled variant of ImageNet as an alternative to the CIFAR datasets. *CoRR*, abs/1707.08819, 2017.

[7] J. Daiber, M. Jakob, C. Hokamp, and P. N. Mendes. Improving efficiency and accuracy in multilingual entity extraction. In *I-SEMANTICS 2013 - 9th International Conference on Semantic Systems, ISEM '13, Graz, Austria, September 4-6, 2013*, pages 121–124. ACM, 2013.

[8] X. Dong et al. More is less: A more complicated network with less inference complexity. In *Proc. CVPR 2017*, pages 1895–1903, Honolulu, HI, USA, July 2017.

[9] K. Fukunaga. *Introduction to statistical pattern recognition (2nd ed.)*. Academic Press Professional, Inc., San Diego, CA, USA, 1990.

[10] S. Ge et al. Compressing deep neural networks for efficient visual inference. In *Proc. ICME 2017*, pages 667–672, Hong Kong, China, July 2017.

[11] N. Gkalelis and V. Mezaris. Subclass deep neural networks: Re-enabling neglected classes in deep network training for multimedia classification. In *Proc. Int. Conf. MultiMedia Modeling*, volume 11961, pages 227–238, Daejeon, South Korea, July 2020.

[12] N. Gkalelis, V. Mezaris, I. Kompatsiaris, and T. Stathaki. Mixture subclass discriminant analysis link to restricted gaussian model and other generalizations. *IEEE transactions on neural networks and learning systems*, 24(1):8–21, 2012.

[13] A. Graves and J. Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18(5):602–610, Mar. 2005.

[14] K. He et al. Deep residual learning for image recognition. In *Proc. CVPR 2016*, pages 770–778, Las Vegas, NV, USA, June 2016.

[15] Y. He, X. Dong, G. Kang, Y. Fu, C. Yan, and Y. Yang. Asymptotic soft filter pruning for deep convolutional neural networks. *IEEE transactions on cybernetics*, 2019.

[16] Y. He et al. Soft filter pruning for accelerating deep convolutional neural networks. In *Proc. IJCAI 2018*, pages 2234–2240, Stockholm, Sweden, July 2018.

[17] Y. He et al. Filter pruning via geometric median for deep convolutional neural networks acceleration. In *Proc. CVPR 2019*, Long Beach, CA, USA, June 2019.

[18] Y. He, X. Zhang, and J. Sun. Channel pruning for accelerating very deep neural networks. In *Proc. ICCV 2017*, pages 1398–1406, Venice, Italy, Oct. 2017.

[19] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[20] J. Hoffart, M. A. Yosef, I. Bordino, H. Fürstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum. Robust disambiguation of named entities in text. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 782–792, 2011.

[21] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[22] A. Joly and O. Buisson. Logo retrieval with a contrario visual query expansion. In *Proceedings of the 17th ACM international conference on Multimedia*, pages 581–584. ACM, 2009.

[23] Y. Kalantidis, L. G. Pueyo, M. Trevisiol, R. van Zwol, and Y. Avrithis. Scalable triangulation-based logo recognition. In *Proceedings of the 1st ACM International Conference on Multimedia Retrieval*, page 20. ACM, 2011.

[24] J. Krause, M. Stark, J. Deng, and L. Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013.

[25] A. Krizhevsky. Learning multiple layers of features from tiny images. Technical report, Department of Computer Science, University of Toronto, 2009.

[26] J. Lee, A. (Paul) Natsev, W. Reade, R. Sukthankar, and G. Toderici. The 2nd YouTube-8M large-scale video understanding challenge. In *ECCV Workshops*, Munich, Germany, September 2018.

[27] H. Li et al. Pruning filters for efficient convnets. In *Proc. ICLR 2017*, Toulon, France, Apr. 2017.

[28] L. Li, J. Zhu, and M. Sun. Deep learning based method for pruning deep neural networks. In *Proc. ICMEW 2019*, pages 312–317, Shanghai, China, July 2019.

[29] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[30] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[31] V. Mezaris, K. Apostolidis, and L. Nixon. ReTV Deliverable 1.1: Data Ingestion, Analysis and Annotation. Technical report, 2018.

[32] V. Mezaris, K. Apostolidis, and L. Nixon. ReTV Deliverable 1.1: Data Ingestion, Analysis and Annotation. Technical report, 2019.

[33] P. Molchanov et al. Pruning convolutional neural networks for resource efficient inference. In *Proc. ICLR 2017*, Toulon, France, Apr. 2017.

[34] A. Moro, A. Raganato, and R. Navigli. Entity linking meets word sense disambiguation: a unified approach. *Transactions of the Association for Computational Linguistics*, 2:231–244, 2014.

[35] A. G. Nuzzolese, A. L. Gentile, V. Presutti, A. Gangemi, D. Garigliotti, and R. Navigli. Open knowledge extraction challenge. In *Semantic Web Evaluation Challenges*, pages 3–15. Springer, 2015.

[36] A. G. Nuzzolese, A. L. Gentile, V. Presutti, A. Gangemi, R. Meusel, and H. Paulheim. The second open knowledge extraction challenge. In H. Sack, S. Dietze, A. Tordai, and C. Lange, editors, *Semantic Web Challenges - Third SemWebEval Challenge at ESWC 2016, Heraklion, Crete, Greece, May 29 - June 2, 2016, Revised Selected Papers*, volume 641 of *Communications in Computer and Information Science*, pages 3–16. Springer, 2016.

[37] K. Ota, M. S. Dao, V. Mezaris, and F. G. D. Natale. Deep learning for mobile multimedia: A survey. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 13(3s):1–22, 2017.

[38] P. Over, G. Awad, M. Michel, J. Fiscus, G. Sanders, B. Shaw, W. Kraaij, A. F. Smeaton, and G. Quéot. Trecvid 2012-an overview of the goals, tasks, data, evaluation mechanisms and metrics. 2013.

[39] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[40] J. Redmon and A. Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[41] M. Röder, R. Usbeck, S. Hellmann, D. Gerber, and A. Both. N^3 - A collection of datasets for named entity recognition and disambiguation in the NLP interchange format. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation, LREC 2014*, pages 3529–3533, 2014.

[42] S. Romberg and R. Lienhart. Bundle min-hashing for logo recognition. In *Proceedings of the 3rd ACM conference on International conference on multimedia retrieval*, pages 113–120. ACM, 2013.

[43] S. Romberg, L. G. Pueyo, R. Lienhart, and R. Van Zwol. Scalable logo recognition in real-world images. In *Proceedings of the 1st ACM International Conference on Multimedia Retrieval*, page 25. ACM, 2011.

[44] J. Salamon and J. P. Bello. Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Processing Letters*, 24(3):279–283, 2017.

[45] H. Su, S. Gong, and X. Zhu. Weblogo-2m: Scalable logo detection by deep learning from the web. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 270–279, 2017.

[46] H. Su, X. Zhu, and S. Gong. Deep learning logo detection with data expansion by synthesising context. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 530–539. IEEE, 2017.

[47] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[48] F. Tafazzoli, H. Frigui, and K. Nishiyama. A large and diverse dataset for improved vehicle make and model recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–8, 2017.

[49] M. Tan and Q. V. Le. Efficientnet: Improving accuracy and efficiency through automl and model scaling. *arXiv preprint arXiv:1905.11946*, 2019.

[50] A. Tüzkö, C. Herrmann, D. Manger, and J. Beyerer. Open set logo detection and retrieval. *arXiv preprint arXiv:1710.10891*, 2017.

[51] P. Warden. Speech commands: A dataset for limited-vocabulary speech recognition. *CoRR*, abs/1804.03209, 2018.

[52] A. Weichselbraun, A. M. Brasoveanu, P. Kuntschik, and L. J. Nixon. Improving named entity linking corpora quality. *RANLP 2019*, pages 1328–1337, 2019.

[53] A. Weichselbraun, A. M. P. Brasoveanu, P. Kuntschik, and L. J. B. Nixon. Improving named entity linking corpora quality. In R. Mitkov and G. Angelova, editors, *Proceedings of the International Conference on Recent Advances in Natural Language Processing, RANLP 2019, Varna, Bulgaria, September 2-4, 2019*, pages 1328–1337. INCOMA Ltd., 2019.

[54] A. Weichselbraun, P. Kuntschik, and A. M. P. Brasoveanu. Name variants for improving entity discovery and linking. In M. Eskevich, G. de Melo, C. Fäth, J. P. McCrae, P. Buitelaar, C. Chiarcos, B. Klimek, and M. Dojchinovski, editors, *2nd Conference on Language, Data and Knowledge, LDK 2019, May 20-23, 2019, Leipzig, Germany.*, volume 70 of *OASICS*, pages 14:1–14:15. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2019.

[55] W. Wen, Y. He, S. Rajbhandari, et al. Learning intrinsic sparse structures within long short-term memory. In *Proc. ICLR*, Vancouver, BC, Canada, 30 Apr.–3 May 2018.

[56] Z. Zhou et al. Online filter weakening and pruning for efficient convnets. In *Proc. ICME 2018*, pages 1–6, San Diego, CA, USA, July 2018.