Deliverable D3.2: Updated Verification Framework (This is a public redacted version of a confidential deliverable.)

Markos Zampoglou, Symeon Papadopoulos, Giorgos Kordopatis-Zilos, Olga Papadopoulou, Vasileios Mezaris, Yiannis Kompatsiaris, Fotini Markatopoulou / CERTH
Lyndon Nixon, Adrian M.P. Brasoveanu / MODUL
Roger Cozien, Gregoire Mercier / EXO MAKINA

22/12/2017

Work Package 3: Content Verification

InVID - In Video Veritas: Verification of Social Media Video Content for the News Industry

Innovation Action

Horizon 2020, Research and Innovation Programme

Grant Agreement Number 687786

Dissemination level	CO
Contractual date of delivery	31/12/2017
Actual date of delivery	22/12/2017
Deliverable number	D3.2
Deliverable name (This is a public redacted version of a confidential deliverable.)	Updated Verification Framework
File	`InVID_D3.2.tex`
Nature	Report
Status & version	Final & V1.0
Number of pages	53
WP contributing to the deliverable	3
Task responsible	CERTH
Other contributors	MODUL, EXO MAKINA
Author(s)	Markos Zampoglou, Symeon Papadopoulos, Giorgos Kordopatis-Zilos, Olga Papadopoulou, Vasileios Mezaris, Yiannis Kompatsiaris, Fotini Markatopoulou / CERTH Lyndon Nixon, Adrian M.P. Brasoveanu / MODUL Roger Cozien, Gregoire Mercier / EXO MAKINA
Quality Assessors	Denis Teyssou / AFP, Max Göbel / WLT
EC Project Officer	Miguel Montarelo Navajo
Keywords	Content verification, video forensic analysis, near-duplicate detection, logo detection, context aggregation, location detection

Abstract

The second year of the InVID project is completed. In the course of the project, we have produced a multitude of tools, integrated in a powerful platform, aimed at assisting journalists to track unfolding events, and collect and verify user-generated video content. WP3 specifically intends to provide a set of tools for news video verification. Building upon our work during the first year (presented in D3.1), in the second year we achieved a number of extensions, modifications and breakthroughs to the WP3 components. The new, updated verification framework presented in this deliverable, matches or exceeds similar attempts from the state of the art, and is already seeing real-world application through its integration to the complete InVID verification tools, namely the Verification Plugin and the Verification Application. Compared to our developments in the first year of the project, all components have substantial improvements. Our work in Video Forensics remains confidential and the corresponding content has been redacted from the document. However, our progress in the other components is presented here openly, taken verbatim from the original, confidential version of D3.2.

– In *Near-Duplicate Detection*, we developed an improved version of the algorithm that surpasses both our algorithm from Year 1, and competing methods from the state of the art. In order to make the Near-Duplicate Detection module useful for analysts, we have began to populate a dataset with a very large number of past videos from breaking events, against which to compare any new incoming videos in order to identify re-use.

– The *Logo Detection* module also underwent a major upgrade by introducing a new, deep learning approach based on an innovative technique for training data generation. The new approach is much faster, and with significant potential for further improvement compared to the version presented in D3.1.

– The *Location Detection* module has significantly increased its performance by improving its popularity and context analysis algorithms, leading to results that surpass the state of the art in most cases. We also present a novel large-scale dataset collected from multiple sources and annotated from multiple perspectives.

– Finally, the *Context Aggregation and Analysis* module was significantly expanded in terms of supported video platforms, improved in terms of speed, and redesigned to present verification-related contextual cues more intuitively.

Overall, the integration of all these analysis components in the InVID platform and tools has progressed smoothly, and for the third year of InVID we intend to further refine these modules and their integration, towards delivering the completed final verification framework at the end of the project.

Content

1 Introduction

This deliverable presents the progress made during the second year of the InVID project for Work Package 3: Content Verification. The objective of WP3 is to develop a set of tools that can assist journalists with content verification tasks, by speeding up existing manual procedures, through innovative and intelligent software components. During Year 2, all the components of WP3 were extended and improved. For each component, this document presents our survey of the state of the art, keeping up-to-date with any recent developments, and describes the new methodologies developed during Year 2. It also documents our evaluations and tangible progress since D3.1, including the effort we put into incorporating feedback from user testing, as well as qualitative and quantitative evaluations on realistic data. It finally documents the current status of integration with the rest of the platform. The only exception is our work in Video Forensics, which remains confidential and has been redacted from this, public version of the document.

The rest of the document is organized as follows: Section 2 presents our analysis of the real-world problem, and the dataset of use-cases we are maintaining, named the Fake Video Corpus. It also analyzes the role of each WP3 component in tackling the problem, their interrelations, and the progress of WP3 as a whole. The following sections are then dedicated to individual components. Section 3 remains as a placeholder for the redacted Video Forensics section, Section 4 deals with Near Duplicate Detection, Section 5 presents the Logo Detection component, Section 6 presents our progress in Location Detection, and Section 7 presents the Context Aggregation and Analysis component. Finally, in Section 8 we provide an outlook of the work done so far and we report our plans for the third year of the project.

1.1 History of the document

Table 1: History of the document

Date	Version	Name	Comment
2017/07/14	V0.1	M. Zampoglou / CERTH	Document structure
2017/08/21	V0.11	S. Papadopoulos, V. Mezaris / CERTH	Structure edits
2017/09/25	V0.2	M. Zampoglou / CERTH	Logo detection section
2017/10/13	V0.3	O. Papadopoulou / CERTH	Context Aggregation and Analysis section
2017/10/17	V0.4	G. Kordopatis-Zilos / CERTH	Near-Duplicate Detection section
2017/10/29	V0.41	V. Mezaris, Y. Kompatsiaris / CERTH	Document structure revisions
2017/11/04	V0.5	L. Nixon, A. Brasoveanu / MODUL	Location Detection section
2017/11/17	V0.6	R. Cozien, G. Mercier / EXO MAKINA	Video Forensics section
2017/12/02	V0.7	F. Markatopoulou, V. Mezaris / CERTH	Video forensics section update
2017/12/03	V0.8	M. Zampoglou, O. Papadopoulou, G. Kordopatis-Zilos, S. Papadopoulos / CERTH	QA version final contributions & revision
2017/12/22	V1.0	M. Zampoglou, S. Papadopoulos, V. Mezaris / CERTH	Final version, ready for submission.
2018/01/08	V1.1	M. Zampoglou, S. Papadopoulos, V. Mezaris / CERTH	Public, redacted version.

1.2 Purpose of the document

The document aims to present our work in WP3 during the second year of InVID. The Work Package contains three tasks:

– **Multimedia forensics**, aiming to detect digital manipulations on the video content by examining the video bitstream (T3.1 - EXO MAKINA). **(REDACTED)**

– **Near-duplicate content detection**, aiming to identify whether a posted image or video has been reposted in the past (T3.2 - CERTH).

– **Contextual verification**, aiming to provide information regarding the location and social network context of a posted item to assist users with verification (T3.3 - CERTH, MODUL).

The purpose of this deliverable is to document the developments for all three of the aforementioned tasks during the second year of the project, and to provide an overall view of the progress achieved towards the WP objectives. The aim of D3.2 is defined as "*...an update of the content verification framework, following its integration and testing on top of the InVID platform. The update will contain both extensions and new implementations, as well as refinements of the previous implementations based on feedback collected from the first evaluation cycles*."

This deliverable presents these extensions and new implementations and their degree of integration with the platform. They are accompanied with qualitative and quantitative evaluations of the achieved performance of the new components, with a focus on progress since Year 1. The achievements of this year include:

1. A significant extension to our real-world video verification dataset, the Video Verification Corpus.
2. An improved near-duplicate video detection algorithm based on CNN features and a learned distance metric designed for improved detection accuracy. Coupled with experiments on fusing multiple classifiers, the new version outperforms both our previous version and the state of the art.
3. A new approach for logo detection, using Region-proposal Convolutional Neural Networks (R-CNNs), leading to increased speed and scalability, and showing greater potential for accuracy improvements.
4. Improvements in the disambiguation process of our location detection module, which led to increased accuracy. A novel, large-scale corpus was also created, for evaluation purposes.
5. Improvements in the context aggregation and analysis module, including extending coverage to other social media platforms, integrating an improved multi-language verification-based comment analysis functionality, integration with the location detection module, as well as speed-ups and adaptations to increase user experience.

In this document, both the status of individual components and of the WP as a whole are presented, and the future steps towards the final year of the project are laid down in the following sections.

1.3 Glossary and Abbreviations

Application Programming Interface (API): In computer programming, an application programming interface (API) is a set of subroutine definitions, protocols, and tools for building application software. In general terms, its a set of clearly defined methods of communication between software components.

Computer Generated Imagery (CGI): This refers to multimedia content (image, video) that is created exclusively or to a large extent with the assistance of software, i.e. does not depict a scene captured from the real world.

Convolutional Neural Networks (CNN): In machine learning, a CNN (or ConvNet) is a type of feed-forward artificial neural network in which the connectivity pattern between its neurons is inspired by the organization of the animal visual cortex. CNNs are typically applied on visual recognition tasks.

Cross-Origin Resource Sharing (CORS): This is a mechanism that allows restricted resources (e.g. fonts) on a web page to be requested from another domain outside the domain from which the resource originated. A web page may freely embed images, stylesheets, scripts, iframes, videos, but certain cross-domain requests, e.g. AJAX requests, are forbidden by default by the same-origin security policy.

Deep Metric Learning (DML): A machine learning approach based on neural networks, where an embedding function is learned to map items to a new feature space based on the pair/triplet-wise relations of the training samples in a development corpus.

Deep Neural Network (DNN): A machine learning model consisting of multiple layers of "artifical neurons" or "units". A modern version of Artificial Neural Networks (ANNs).

Discrete Cosine Transform (DCT): The DCT is a technique for converting a signal into elementary frequency components.

Exchangeable image file format (Exif): This is a standard that specifies the formats for images, sound, and ancillary tags used by digital cameras (including smartphones), scanners and other systems handling image and sound files recorded by digital cameras.

Fake Video Corpus (FVC): The video dataset created within invid for the purposes of identifying and classifying types of fakes, and evaluating various verification approaches.

Image/Video Tampering: This is the act of digitally altering an image or video file either to enhance it (e.g. improve contrast) or to mislead people by generating false evidence. Tampering is also referred to as forgery, manipulation or more colloquially as photoshopping.

International Press Telecommunications Council (IPTC): This is a consortium of the world's major news agencies, other news providers and news industry vendors and acts as the global standards body of the news media. The IPTC defined a set of metadata properties that can be applied to images, part of a broader standard known as the IPTC Information Interchange Model (IIM).

JavaScript Object Notation (JSON): This is an open-standard format that uses human-readable text to transmit data objects consisting of attributevalue pairs. It is the most common data format used for asynchronous browser/server communication.

JPEG: This is a commonly used method of lossy compression for digital images, particularly for those images produced by digital photography. The degree of compression can be adjusted, allowing a selectable trade-off between storage size and image quality. The term "JPEG" is an acronym for the Joint Photographic Experts Group, which created the standard.

MPEG-4: This is a method of defining compression of audio and visual (AV) digital data. It was introduced in late 1998 and designated a standard for a group of audio and video coding formats and related technology agreed upon by the ISO/IEC Moving Picture Experts Group (MPEG).

Named-entity recognition (NER): This is a subtask of information extraction that seeks to locate and classify named entities in text into pre-defined categories such as the names of persons, organizations, locations, expressions of times, quantities, monetary values, percentages, etc.

Near-duplicate detection (NDD), Near-Duplicate Video Retrieval (NDVR): This refers to the task of retrieving multimedia items (images, videos) that are highly similar or identical to a given multimedia item, which is referred to as query.

Radial Basis Function Support Vector Machine (RBF-SVM): An Support Vector Machine is a supervised machine learning model able to achieve non-linear classification through so-called "kernel functions". Radial Basis Functions are a type of such kernel functions.

Region proposal Convolutional Neural Network (RCNN): A type of Deep Neural Network which takes an image as input, and returns a number of region proposals and the classification results for each one of them, thus performing object detection.

Representational state transfer (REST): Also known as RESTful Web services, this refers to a paradigm of providing interoperability between computer systems on the Internet. REST-compliant Web services allow requesting systems to access and manipulate textual representations of Web resources using a uniform and predefined set of stateless operations.

SPARQL Protocol and RDF Query Language (SPARQL): This is an RDF query language, i.e. a semantic query language for databases, able to retrieve and manipulate data stored in Resource Description Framework (RDF) format.

Speeded Up Robust Features (SURF): In computer vision, SURF is a local feature detector and descriptor. It can be used for tasks such as object recognition, image registration and classification.

Strongly Connected Component (SCC): In the mathematical theory of directed graphs, a graph is said to be strongly connected if every vertex is reachable from every other vertex. The strongly connected components of a directed graph form a partition into subgraphs that are themselves strongly connected.

TUNGSTENE: Proprietary software by EXO MAKINA for performing image forensics analysis.

Term Frequency - Inverse Document Frequency (tf-idf): This is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. It is often used as a weighting factor in information retrieval, but also in the context of image retrieval in conjunction with *visual vocabularies*.

Uniform Resource Locator (URL): Commonly termed a web address, this is a reference to a web resource that specifies its location on a computer network and a mechanism for retrieving it.

User Generated Content (UGC): This refers to multimedia content that is generated by any individual (i.e. often amateurs) and is publicly shared through some media sharing platform (e.g. YouTube, Facebook, Twitter, etc.).

Vector of Locally Aggregated Descriptors (VLAD): This is a method of encoding many local descriptors, such as SURF, extracted from an image (or video frame) into a single vector that represents the whole image.

Video Forensics: This refers to a class of video analysis methods that aim to detect traces of tampering in video content.

Work Package (WP): This refers to the structure of InVID work into units called Work Packages.

2 Content verification – Overview

2.1 Content verification in the Wild

In D3.1 we formed a small typology of the various cases of fake[1] videos that may be encountered in the real world. In addition, we presented a first version of the Fake Video Corpus, a dataset consisting of real-world cases of fake videos that were posted in the past, posing as real content. The original corpus consisted of 59 fake videos to be used as a small benchmark for video verification, as well as a demonstration of the various types of fakes.

During the second year, we continued expanding the corpus with more cases, aiming to reach a size which would allow for quantitative evaluations. After the expansion, the Corpus contains 117 fake and 110 real videos. The inclusion of real (verified) videos is meant to allow evaluations against potential false detections (i.e. real videos that are erroneously classified as fake). This set includes verified UGC news videos, the authenticity of many of which was questioned at the time of their posting. Figure 1 shows two sample real and two sample fake videos from the new additions to the Corpus.

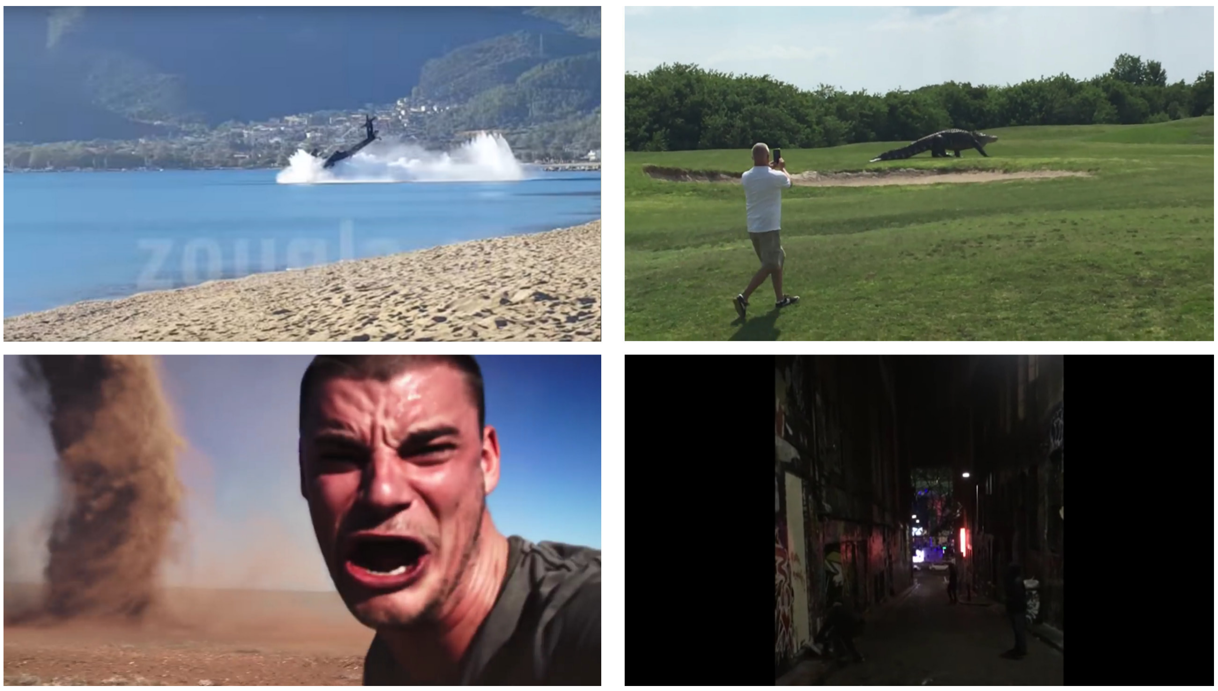


Figure 1: Top: two sample real videos from the extended Fake Video Corpus. Left: a Greek army helicopter crashes into the sea near a beach. Right: a giant alligator walks across a golf course in Florida. Bottom: two sample fake videos. Left: a person running towards a tornado in Australia in order to take a selfie. The video was edited. Right: the artist Banksy caught on camera while making a graffiti. The video was staged.

2.2 Content verification in InVID

In D3.1 we presented the WP3 modules, and how each module was related to our definition and analysis of "Fake" videos. During the second year of the project, the structure and the role of each component have largely remained the same, despite various important improvements on the individual components, their relations, and the overall framework coverage.

The Video Forensics component aims at detecting digitally tampered videos. However, Section 3 describing our progress in the field has been redacted for reasons of confidentiality.

[1]While the term "fake", as in "fake news" is very popular, it may be misleading in light of the complexity of the problem of mis- and disinformation. In this document we opted to use the term "fake" for its simplicity and recognizability, but the reader should be aware that the issue is much more multi-faceted. E.g. see `https://medium.com/1st-draft/fake-news-its-complicated-d0f773766c79`

The Near Duplicate Detection module, on the other hand, aims to detect videos that originate from past events and are being posted as being captured in the context of an ongoing event. In Section 4 we present our progress in improving the speed and accuracy of the system, as well as building a large dataset of videos with which to compare each new query, so as to increase the probability of identifying a case of re-use.

The Logo Detection module is also aimed at assisting investigators to contextually verify videos, by identifying who has posted (or re-posted) it. While not directly used for verification, identifying the provenance of the video can alert the journalist to be wary of potential biases. This in turn may be helpful in knowing where the falsehood may lie, and which aspect of the video content or context to focus on during the verification process. In Section 5 we present the new Logo Detection algorithm and the increased speed and scalability it provides with minimal loss in accuracy.

The Location Detection module is similar to Logo Detection, in the sense that it does not provide direct assessments on a video, but rather contextual cues that can be used for verification by human users. In Section 6 we present our progress in this module, including the creation of a very large dataset and significant improvements on algorithm accuracy.

Finally, the Context Aggregation and Analysis (CAA) module attempts to provide a tool that can be helpful in identifying all cases of forgeries by holistically analyzing their context, from location, to weather, to user comments. In Section 7 we present the expansions and improvements on that module.

During the second year, the improvements and adaptations of all these modules were based on user testing, feedback and our own observations. One change that was common across multiple modules (Logo Detection, Near Duplicate Detection, Context Aggregation and Analysis) was the extension of their coverage to more platforms. While in the first year, the majority of modules were only applicable on YouTube videos, following the observation that a large amount of news-related content is shared through other media platforms (e.g. Twitter, Facebook), these modules were extended to work on videos uploaded on these platforms as well. Another change concerning the synergy between modules was the integration of the Location Detection module with the Context Aggregation and Analysis, and the replacing of the previous, keyword-based location detection field of CAA. Since both modules provide contextual information, the overlap was removed, and the advanced localization capabilities of the Location Detection module are now used for contextual verification. Details on these changes are presented in the CAA section.

2.2.1 Progress and evaluations during Year 2

Besides these overarching changes, all modules underwent significant improvements in their features, underlying algorithms, and service implementations. To evaluate the improvements in performance and resulting user experience, both quantitative and qualitative evaluations were run during the second project year. With respect to quantitative evaluations, the algorithms developed for each component were evaluated on task-specific datasets. The Fake Video Corpus remains a central dataset for the work package, both as a demonstration of different types of "fake" and as an evaluation benchmark where appropriate, however it only consists of a small number of videos covering all different types of fakes. While it remains central for WP3, we realized early on the need for more specialized datasets, and had began collecting them since the first year. Thus, at this stage, besides the FVC, the following datasets have been created for InVID and have been used in the evaluations presented in this document[2]:

– A Near-Duplicate Detection dataset comprising: a) videos from current events collected through InVID's Story Detection module, and b) videos from past events gathered through search queries. The dataset is currently being annotated and manually filtered to serve as a benchmark for near-duplicate detection evaluations, containing many near-duplicates in the form of modified copies and edits, but also videos of the same event from different angles, and a large number of distractor videos. The dataset is also important for the usefulness of the module, as any new videos are compared against this collection, thus, it has to be as extensive as possible to catch cases of re-use. The dataset is presented in Section 4.

– A Logo Detection dataset consisting of a large number of TV videos, segmented into shots and annotated by the logos they contain. The dataset was also used in the first year's evaluations of the Logo Detection algorithm, and is used again in the evaluations of Section 5

[2]More details regarding the data management aspects of these datasets are provided in the updated Data Management Plan.

– A dataset that was initially created in order to improve the geolocation functionality of the Recognyze tool but was subsequently expanded in order to cover events and other entity types as well. It contains documents from multiple types of sources (e.g. tweets, subtitles, news articles) collected during the early months of the Summer 2017, annotated using multiple approaches ("Lenses"). It is presented in Section 6.

Besides these datasets that we created ourselves, established benchmark datasets were also used for quantitative evaluations of the various modules, such as the Video Copy DataBase (VCDB) dataset and the CC WEB VIDEO dataset used for near-duplicate detection, and the LDL-2016 and N3 Corpora datasets used in location detection evaluation. Furthermore, we tried to use the Fake Video Corpus for evaluations wherever it was relevant -in the public version of this document, it is used for the quantitative experiments of the Context Aggregation and Analysis component of Section 7, where the varied nature of the dataset's contents makes it most suitable for our evaluations.

Finally, another important part in the evaluation of the delivered components was based on the four conducted InVID test and validation cycles[3]. The results of these repetitive technology evaluation procedures, and the impact they had in the improvement of the components, are presented in the corresponding sections.

[3]While the first test and validation cycle technically took place during the first project year, we were not able to incorporate its results in D3.1 since it closed in mid-M12. Thus, its results are presented here. Similarly, test and validation cycle 5 is expected to conclude in mid-M24, and so its results will be included in D3.3.

3 Video Forensics

[Content removed as confidential.]

4 Near-duplicate Detection

During the second project year, we dedicated effort on improving the retrieval performance of the Near-Duplicate Detection (NDD) approach, further surpassing the current state of the art in accuracy. We achieved this by developing a Deep Metric Learning approach which allows us to use learned distance measures to evaluate similarity. Additionally, we also improved the component through integration of all contributions into the InVID platform, and fixing problems or bugs that were identified during the project test cycles.

4.1 State of the art

A thorough study on the Near-Duplicate Video Retrieval (NDVR) problem and several recent approaches is provided by Liu et al. (J. Liu et al., 2013). According to it, existing NDVR methods are classified based on the granularity of the matching between NDVs into video-, frame- and hybrid-level matching.

Video-level matching: These approaches aim at solving the NDVR problem at massive scale. Videos are usually represented with a global signature such as an aggregate feature vector (X. Wu, Hauptmann, & Ngo, 2007; L. Liu, Lai, Hua, & Yang, 2007; Z. Huang, Shen, Shao, Zhou, & Cui, 2009) or a hash code (Song, Yang, Huang, Shen, & Hong, 2011; Hao et al., 2017; Song, Yang, Huang, Shen, & Luo, 2013) and the video matching is based on the computation of the pairwise similarity between the corresponding video representations.

Frame-level matching: NDVs are determined in this case by comparing between individual frames or frame sequences of the candidate videos. Existing approaches (Douze, Jegou, & Schmid, 2010; Cai et al., 2011) calculate frame-by-frame similarity based on Bag-of-Words (BoW) schemes or employ sequence alignment algorithms. Other works have explored spatio-temporal representations (Shang, Yang, Wang, Chan, & Hua, 2010; Zhang, Ren, Chang, Wood, & Kender, 2012) to improve retrieval performance and accelerate the similarity computation.

Hybrid-level matching: Such approaches attempt to combine the advantages of video- and frame-level methods. Typical such approaches are, for instance, presented in (X. Wu et al., 2007; Chou, Chen, & Lee, 2015), both of which first employ a filter-and-refine scheme to cluster and filter out near-duplicate videos, and then use frame-to-frame similarity on the reduced set of videos.

Another field of study relevant to our work is metric learning, for which a detailed survey is provided by Yang and Jin (Yang, 2006). Metric learning is conducted using pairwise (Hadsell, Chopra, & LeCun, 2006; Zheng, Gong, & Xiang, 2011; Mignon & Jurie, 2012; Radenović, Tolias, & Chum, 2016) or triplet-wise constraints (Chechik, Sharma, Shalit, & Bengio, 2010; P. Wu et al., 2013; Wang et al., 2014; Schroff, Kalenichenko, & Philbin, 2015; Chen, Yuan, Hua, Zheng, & Wang, 2015). Its main purpose is to learn an optimal projection for mapping input features to another feature space. In the case of NDVR, we aim at an embedding function that maps NDVs closer to each other than to the rest of the videos.

Pairwise methods usually employ contrastive loss that tries to minimize the distance between pairs of examples with same-class labels, while penalizing examples with different-class labels that are closer than a margin γ (Hadsell et al., 2006; Radenović et al., 2016). Triplet-wise embedding is trained on triplets of data with an anchor point, a positive that belongs to the same class, and a negative that belongs to a different class (Wang et al., 2014; Schroff et al., 2015; Chen et al., 2015). Triplet-wise methods use a loss over triplets to push the anchor and positive close, and penalize triplets where the distance between the anchor and the negative is less than the one between the anchor and the positive plus a margin γ. Deep metric learning has been successfully applied to a variety of problems including image retrieval (P. Wu et al., 2013; Wang et al., 2014; Radenović et al., 2016), face recognition/retrieval (Schroff et al., 2015), person re-identification (Paisitkriangkrai, Shen, & van den Hengel, 2015), etc.

4.2 Method description

During the second project year, effort was placed on: a) improving the performance of the approach for more efficient retrieval of near-duplicates, b) updating the service to integrate the new approach, support any video platform and provide near-duplicate localization, c) implementing all the recommendations and bug fixes submitted during the test cycles as part of WP7 and d) expanding the database of indexed videos by collecting content from the most important events that occurred in the last five years. Finally, an independent annotation tool was developed for manual annotation of near-duplicate videos, that will be used for the generation of an evaluation dataset.

In the D3.1 we developed an NDVR approach that leverages features produced by the intermediate convolution layers of deep CNN architectures. We aggregate the produced layer features to a video representation based on an scheme that utilize multiple codebooks. The video similarity was determined by the cosine similarity between the video representations. Given a query, a number of candidate videos was retrieved from an inverted file structure, ranked using their similarity to the query.

To improve the performance, we focused on the development of a learning approach based on Deep Metric Learning (DML) (Kordopatis-Zilos, Papadopoulos, Patras, & Kompatsiaris, 2017b). To this end, we have built an triplet-wise architecture to learn an embedding function that maps videos to a feature space where near-duplicate videos have smaller distances between each other compared to other videos. Moreover, two different fusion variations have been tested for the generation of the video representation. The generated video representation is compact in order to facilitate the development of scalable NDVR systems.

For feature extraction, we adopt a similar procedure to the one described in D3.1 (Kordopatis-Zilos, Papadopoulos, Patras, & Kompatsiaris, 2017a). More precisely, a CNN pre-trained on ImageNet (Deng et al., 2009a) [4] is employed to extract frame descriptors. Hence, given an input image to the CNN, Maximum Activation of Convolutions (MAC) (Razavian, Sullivan, Carlsson, & Maki, 2016) is applied to each intermediate convolutional layer. The results of the MAC application to each layer are concatenated to a single descriptor. Finally, the frame descriptors are normalized by applying zero-mean and ℓ_2-normalization. To generate global video descriptors, uniform sampling is initially applied to select one frame per second for every video and extract the respective features for each of them. Global video descriptors are then derived by averaging and normalizing (zero-mean and ℓ_2-normalization) these frame descriptors.

We address the problem of learning a pairwise similarity function for NDVR from the relative information of pair/triplet-wise video relations. For a given query video and a set of candidate videos, the goal is to compute the similarity between the query and every candidate video and use it for ranking the entire set of candidates in the hope that the near-duplicates are retrieved at the top ranks. To formulate this process, we define the similarity between two arbitrary videos q and p as the squared Euclidean distance in the video embedding space (Equation 1).

$$\mathsf{D}(f_\theta(q), f_\theta(p)) = \|f_\theta(q) - f_\theta(p)\|_2^2 \tag{1}$$

where $f_\theta(\cdot)$ is the embedding function that maps a video to a point in an Euclidean space, θ are the system parameters and $\mathsf{D}(\cdot,\cdot)$ is the squared Euclidean distance in this space.

Our objective is to learn an embedding function $f_\theta(\cdot)$ that assigns smaller distances to NDV pairs compared to non-NDV ones. Given a video with feature vector v, a NDV with v^+ and a dissimilar video with v^-, the embedding function $f_\theta(\cdot)$ should map video representations to a common space $\mathbb{R}^d$, where d is the dimension of the feature embedding, in which the distance between query v and positive v^+ is always smaller than the distance between query v and negative v^- (Equation 2).

$$\begin{gathered}\mathsf{D}(f_\theta(v), f_\theta(v^+)) < \mathsf{D}(f_\theta(v), f_\theta(v^-)),\\ \forall v, v^+, v^- \text{ such that } \mathsf{I}(v, v^+) = 1, \mathsf{I}(v, v^-) = 0\end{gathered} \tag{2}$$

where, $\mathsf{I}(\cdot,\cdot)$ is a pairwise indicator function, which specifies whether a pair of videos are near-duplicate.

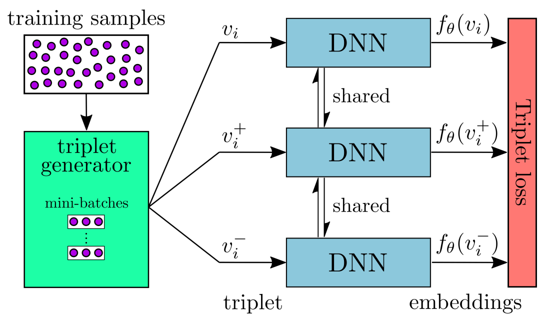
To implement the learning process, we create a collection of N training instances organized in the forms of triplets $\mathscr{T} = \{(v_i, v_i^+, v_i^-), i = 1, ..., N\}$, where v_i, v_i^+, v_i^- are the feature vectors of the query, positive (NDV), and negative (dissimilar) videos. A triplet expresses a relative similarity order among three videos, i.e. v_i is more similar to v_i^+ in contrast to v_i^-. We define the following hinge loss function for a given triplet called 'triplet loss' (Equation 3).

$$L_\theta(v_i, v_i^+, v_i^-) = \max\{0, \mathsf{D}(f_\theta(v_i), f_\theta(v_i^+)) - \mathsf{D}(f_\theta(v_i), f_\theta(v_i^-)) + \gamma\} \tag{3}$$

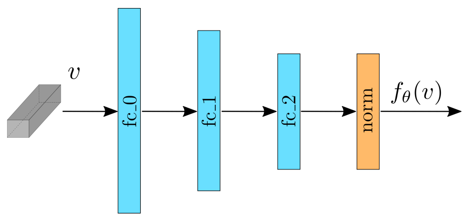
where γ is a margin parameter to ensure a sufficiently large difference between the positive-query distance and negative-query distance. If the video distances are calculated correctly within margin γ, then this triplet will not be penalised. Otherwise the loss is a convex approximation that measures the degree of violation of the desired distance between the video pairs specified by the triplet. To this end, we use batch gradient descent to optimize the objective function described in Equation 4.

$$\min_\theta \sum_{i=1}^{m} L_\theta(v_i, v_i^+, v_i^-) + \lambda \|\theta\|_2^2 \tag{4}$$

[4] http://www.image-net.org/



(a) DML architecture



(b) DNN

Figure 2: Illustration of (a) the DML architecture, and (b) the composition of the DNN.

where λ is a regularization parameter to prevent overfitting of the model, and m is the total size of a triplet mini-batch. Minimising this loss will narrow the query-positive distance while widening the query-negative distance, and thus lead to a representation satisfying the desirable ranking order. With an appropriate triplet generation strategy in place, the model will eventually learn a video representation that improves the effectiveness of the NDVR solution.

For training the DML model, a siamese deep network architecture is utilized (Figure 2(a)) that optimizes the triplet loss function of Equation 3. The network is provided with a set of triplets $\mathscr{T}$ created by the triplet generation process. Each triplet contains a query, a positive and a negative video with v_i, v_i^+ and v_i^- feature vectors, respectively, which are fed independently into three siamese Deep Neural Networks (DNNs) with identical architecture and parameters. The DNNs compute the embeddings of $v : f_\theta(v) \in \mathbb{R}^d$. The architecture of the deployed DNNs is based on three dense *fully-connected layers* and a *normalization layer* at the end leading to vectors that lie on a d-dimensional unit length hypersphere, i.e. $\|f_\theta(v)\|_2 = 1$ (Figure 2(b)). The size of each hidden layer (number of neurons) and the d-dimension of the output vector $f_\theta(v)$ depends on the dimensionality of input vectors, which is in turn dictated by the employed CNN architecture. The video embeddings computed from a batch of triplets are then given to a triplet loss layer to calculate the accumulated cost based on Equation 3.

After training, the learned embedding function is used for computing similarities between videos in a target video corpus. Two variants are proposed for fusing similarity computation across video frames (Figure 3): (i) Early fusion: frame descriptors are averaged and normalized into a global video descriptor, before they are forward propagated to the network. The global video signature is the output of the embedding function $f_\theta(\cdot)$, and (ii) Late fusion: Every extracted frame descriptor of an input video is fed forward to the network, and the set of their embedding transformations is averaged and normalized.

There are several pros and cons for each scheme. The former is computationally lighter and more intuitive; however, it is slightly less effective. Late fusion leads to better performance and is amenable to possible extensions of the base approach (i.e. frame-level approaches). Nonetheless, it is slower since the features extracted from all selected video frames are fed to the DNN.

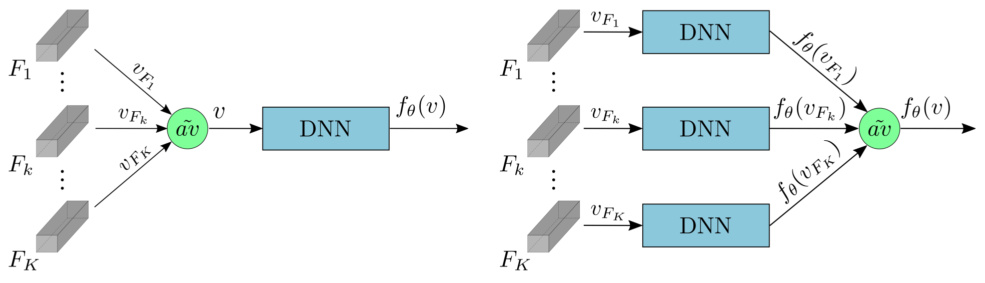
Finally, the similarity between two videos derives from the distance of their representations. For a given query q and a set of M candidate videos $\{p_i\}_{i=1}^M \in P$, the similarity within each candidate pair is determined by Equation 5.

$$\mathsf{S}(q,p) = 1 - \mathsf{D}(f_\theta(q), f_\theta(p)) / \max_{p_i \in P}(\mathsf{D}(f_\theta(q), f_\theta(p_i))) \tag{5}$$

where $S(\cdot,\cdot)$ is the similarity between two videos and $\max(\cdot)$ is the maximum function.

Finally, a critical component of the approach is the generation of the video triplets. It is important to provide a considerable amount of videos for constructing a representative triplet training set. However, there is a massive number of triplets that can be generated. We have empirically determined that only a tiny portion of videos in a video corpus could be considered as near-duplicates for a given video query. Thus, it would be inefficient to randomly select video triplets from this vast set. Instead, a sampling strategy is employed as a key element of the triplet generation process, which is focused on selecting hard candidates to create triplets.

The proposed sampling strategy is applied on a development dataset. Such a dataset needs to contain two sets of videos: $\mathscr{P}$, a set of near duplicate video pairs that are used as query-positive pairs, and $\mathscr{N}$, a set of dissimilar videos that are used as negatives. We aim at generating *hard triplets*, i.e. negative videos (*hard negatives*) with distance to the query that is smaller than the distance between



(a) Early fusion (b) Late fusion

Figure 3: Illustration of early and late fusion schemes.

the query and positive videos (*hard positives*). This condition is expressed in Equation 6.

$$\mathscr{T} = \{(q,p,n)|(q,p) \in \mathscr{P}, n \in \mathscr{N}, \mathsf{D}(q,p) > \mathsf{D}(q,n)\} \tag{6}$$

where $\mathscr{T}$ is the resulting set of triplets. The global video features are first extracted following the feature extraction process. Then, the distance between every query in $\mathscr{P}$ and every dissimilar video in $\mathscr{N}$ is calculated. If the query-positive distance is greater than a query-negative distance, then a hard triplet is formed composed by the three videos. The distance is calculated based on the Euclidean distance of the initial global video descriptors.

4.3 Evaluation and progress since Year 1

4.3.1 Dataset expansion

During the second year of the project, we dedicated effort to building a large news-related video dataset for near-duplicate retrieval. This is a major prerequisite for the service to be of use to journalists, since in the case of fake videos taken from past events, near-duplicate detection can only work as a verification tool if the original video is already present in the dataset. Thus, we need to build a video database that is as large as possible, to increase the likelihood of retrieving a match given a query video that has been sourced from a previously published video.

To build a rich database of news related videos, there are two distinct video sources. The first one includes videos that come from events that are unfolding in the present or the very recent past. There is a plethora of video content that derives from breaking-news events. The second source comprises videos that are related to past events. A comprehensive database has to contain archive/historic videos from the major events that occurred in the past years, because these are videos that are often reused.

To expand our database with videos from the current events, the NDD service has been connected with the automatic crawler developed for story detection from WP2. Every video that is collected by the crawler is automatically sent to the NDD service for indexing. In this way, approximately 1,200 breaking-news videos are added to the video index daily.

For past events, we have set up a workflow to retrieve videos from the major events of the recent years, as follows: First, we crawled Wikipedia's "Current Event" page[5] to build a collection of the major events of the last years. Each event is associated with a topic, headline, text, date, and hyper-references. Then, all events with topic related to 'Armed conflicts and attacks' or 'Disasters and accidents' were retained. The open APIs of YouTube, Dailymotion, New York Times and The Guardian were used to collect videos and articles by providing the events' headlines as queries. The results were filtered to contain only videos published after the corresponding event's start date and up to one week after that date. Furthermore, they were filtered to contain only videos whose duration did not exceed five minutes.

The time interval used for the crawling of the events was from January 2013 to August 2017. A total of 8,467 events were collected, and 4,433 events were retained after the filtering. From the querying of the media platforms we collected: 267,148 videos from Youtube (∼60 videos/event), 820,722 from Dailymotion (∼185 videos/event), 6,864 from New York Times (∼1.6 articles/event) and 91,218 from The Guardian (∼21 articles/event). However, we empirically found that the Dailymotion and The Guardian APIs return many items unrelated to the query, so their results need to be filtered further before being used in evaluations. Nonetheless, it is a rich database, with multiple near-duplicate videos that are

[5] `https://en.wikipedia.org/wiki/Portal:Current_events`

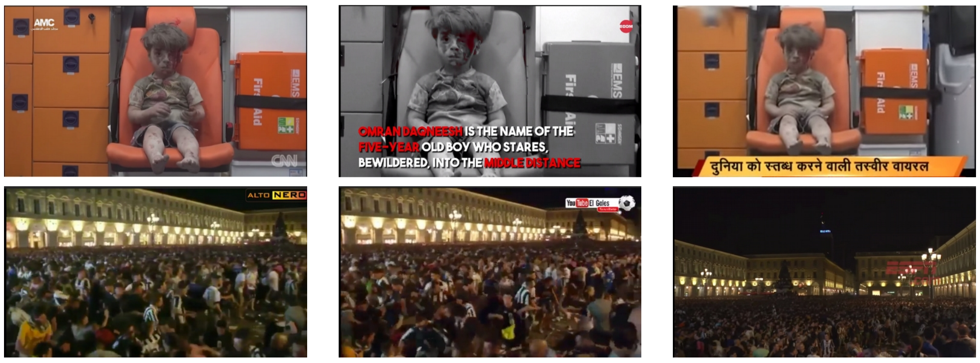

OMRAN DAQNEESH IS THE NAME OF THE FIVE-YEAR OLD BOY WHO STARES, BEWILDERED, INTO THE MIDDLE DISTANCE
दुनिया को स्तब्ध करने वाली तस्वीर वायरल

Figure 4: Examples of video frames from the collected videos.

heavily modified, videos depicting the same event from different angles, and many distractor videos that may fool many NDD algorithms by appearing similar to queries without being near-duplicates. Some examples are illustrated in Figure 4.

4.3.2 Evaluation

For the training of the model, we leverage the Video Copy DataBase (VCDB) dataset (Jiang, Jiang, & Wang, 2014) to generate triplets for training our DML-based system. This dataset is composed of videos from popular video platforms and has been compiled and annotated as a benchmark for the partial copy detection problem. VCDB contains two subsets: (i) the core $\mathscr{C}_c$ with 528 query videos and over 9,000 pairs of partial copies, and (ii) the distractor subset $\mathscr{C}_d$ with 100,000 distractor videos that is used to make the video copy detection problem more challenging. For the triplet generation, we retrieve all video pairs that have been annotated as partial copies. We define an overlap criterion that determines whether a pair is going to be used for the triplet generation: if the duration of the overlap content is greater than a certain threshold t compared to the total duration of each video, then the pair is retained; otherwise, it is discarded. Each video of a given pair can be used once as query and once as positive video. Therefore, the set of query-positive pairs $\mathscr{P}$ is generated based on Equation 7.

$$\mathscr{P} = \{(q,p) \cup (p,q) | q,p \in \mathscr{C}_c, \mathrm{o}(q,p) > t\} \tag{7}$$

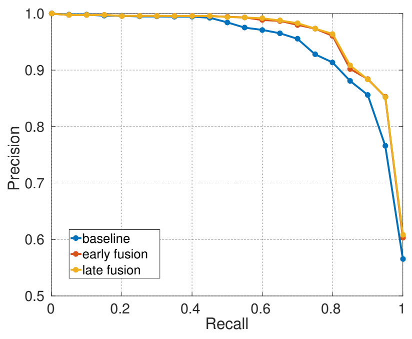
where $\mathrm{o}(\cdot,\cdot)$ determines the video overlap. We found empirically that the selection of the threshold t has considerable impact on the quality of the resulting DML model. Subset $\mathscr{C}_d$ is used as the set $\mathscr{N}$ of negatives. To generate hard triplets, the negative videos are selected from $\mathscr{C}_d$ based on Equation 6.

Similar to D3.1, experiments were performed on the CC_WEB_VIDEO dataset (X. Wu et al., 2007) that consists of 24 queries with a total of 13,129 videos and 397,965 keyframes. For the needs of our approach, we extracted one frame per second for every video in the dataset resulting in a total of approximately 2M video frames. Additionally, for the evaluation of the approach we use the interpolated precision-recall (PR) curve and the mean average precision (mAP) across all queries.

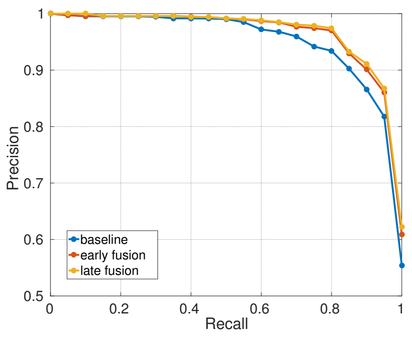
We experimented with two deep network architectures: AlexNet (Krizhevsky, Sutskever, & Hinton, 2012) and GoogleNet (Szegedy et al., 2015). For the former, all convolution layers were used for the extraction of the frame descriptors, whereas, for the latter, all inception layers. The generated vectors have 1,376 and 5,488 dimensions respectively. Both architectures receive images of size 224×224 as input (input frames were resized to these dimensions). For feature extraction, we used the Caffe framework (Jia et al., 2014), which provides pre-trained models on ImageNet for both employed CNN networks[6]. The implementation of the deep model was based on Theano (Theano Development Team, 2016). For the three hidden layers [`fc_0`, `fc_1`, `fc_2`], we used [800, 400, 250] and [2500, 1000, 500] neurons for AlexNet and GoogleNet respectively. Thus, the dimensionality of the output embeddings was 250 and 500 dimensions for the two architectures respectively. Adam optimization (Kingma & Ba, 2014) was employed with learning rate $l = 10^{-5}$ and mini-batches of size $m = 1000$ triplets. For the triplet generation, we set $t = 0.8$, which generates approximately 2k pairs in $\mathscr{P}$ and 7M and 5M triplets in $\mathscr{T}$, for AlexNet and GoogleNet, respectively. Other parameters were set to $\gamma = 1$ and $\lambda = 10^{-5}$.

The performance of the approach in the CC_WEB_VIDEO dataset for the two architecture (AlexNet and GoogleNet) is illustrated in Figure 5 and Table 2. For each of architectures, three configurations

[6]`https://github.com/BVLC/caffe/wiki/Model-Zoo`



(a) AlexNet



(b) GoogleNet

Figure 5: Precision-Recall curve of the proposed approach based on the two CNN architectures and for the three system setups.

were tested: i) **baseline**: fuse all frame descriptors to a single vector and use it for retrieval without any transformation, ii) **early fusion**: fuse all frame descriptors to a single vector and then apply the learned embedding function to generate the video descriptor for retrieval, iii) **late fusion**: apply the learned embedding function to every frame descriptor and fuse the embeddings to derive video representations for retrieval. Late fusion runs outperformed both baseline and early fusion ones for both CNN architectures. GoogleNet achieved better results for all three settings with considerable margin, with precision more than 97% up to 80% recall and mAP scores of 0.968 and 0.969 for early and late fusion respectively. Both fusion schemes clearly improved the performance of the baseline approach for both architectures. Both schemes achieve very similar results, which indicates that the choice of the employed fusion scheme is not crucial for the performance of the method.

Architecture	baseline	early fusion	late fusion
AlexNet	0.948	0.964	0.964
GoogleNet	0.952	0.968	**0.969**

Table 2: mAP of both CNN architectures based on the baseline and two DML fusion schemes.

The proposed approach was compared against six approaches from the literature. Four of those were developed having access to the evaluation set. The remaining two do not require a development dataset.

Auto Color Correlograms (ACC): Cai et al. (Cai et al., 2011) use uniform sampling to extract one frame per second for the input video. The auto-color correlograms (J. Huang, Kumar, Mitra, Zhu, & Zabih, 1999) of each frame are computed and aggregated based on a visual codebook generated from a training set of video frames. The retrieval of near-duplicate videos is performed using tf-idf weighted cosine similarity over the visual word histograms of a query and a dataset video.

Pattern-based approach (PPT): Chou et al. (Chou et al., 2015) build a pattern-based indexing tree (PI-tree) based on a sequence of symbols encoded from keyframes, which facilitates the efficient retrieval of candidate videos. They use m-pattern-based dynamic programming (mPDP) and time-shift m-pattern similarity (TPS) to determine video similarity.

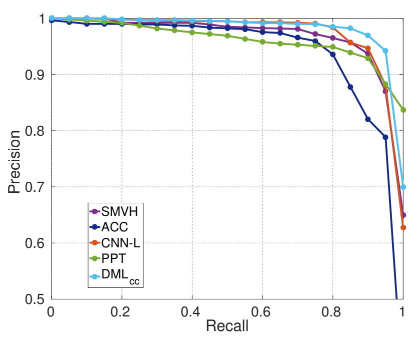
Stochastic Multi-view Hashing (SMVH): Hao et al. (Hao et al., 2017) combine multiple keyframe features to learn a group of mapping functions that project video keyframes into the Hamming space. The combination of keyframe hash codes generates a video signature that constitutes the final video representation. A composite KL divergence measure is used to compute similarity scores.

Layer-wise Convolutional Neural Networks (CNN-L): we also compared the approach with the one used in D3.1 (Kordopatis-Zilos et al., 2017a) using GoogleNet for feature extraction.

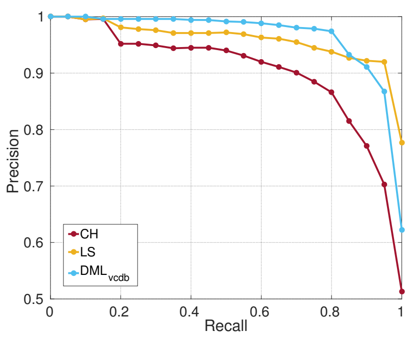
The remaining two approaches are based on the work of Wu et al. (X. Wu et al., 2007):

Color Histograms (CH): This is a global video representation based on the color histograms of keyframes. The color histogram is a concatenation of 18 bins for Hue, 3 bins for Saturation, and 3 bins for Value, resulting in a 24-dimensional vector representation for every keyframe. The global video signature is the normalized color histogram over all keyframes in the video.

Local Structure (LS): Global signatures and local features are combined using a hierarchical approach.



(a) Evaluation dataset



(b) No/Other dataset

Figure 6: Precision-Recall curve of the proposed approach and state-of-the-art approaches, separated by the development dataset.

	Evaluation Dataset					No/Other Dataset		
Method	**ACC**	**PPT**	**SMVH**	**CNN-L**	**DML$_{cc}$**	**CH**	**LS**	**DML$_{vcdb}$**
mAP	0.944	0.958	0.971	0.974	**0.981**	0.892	0.954	**0.969**

Table 3: mAP comparison between two variants of the proposed approach against six state-of-the-art methods. The approaches are divided based on the dataset used for development.

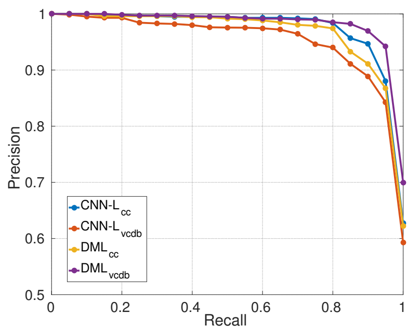
Color signatures are employed to detect near-duplicate videos with high confidence and to filter out very dissimilar videos. For the reduced set of candidate videos, a local feature based method was developed, which compares the keyframes in a sliding window using their local features (PCA-SIFT (Ke & Sukthankar, n.d.)).

For comparing the performance of our approach with the six NDVR approaches from the literature, we selected the setup using GoogleNet features and late fusion denoted as DML$_{vcdb}$, since it achieved the best results. For the sake of comparison and completeness, we further provide the results of our model trained on a triplet set derived from both VCDB (similar to DML$_{vcdb}$) and also videos sampled from CC_WEB_VIDEO, denoted as DML$_{cc}$. The latter simulates the situation where the DML-based approach had access to a portion of the evaluation corpus, similar to the setting used by the competing approaches. Table 3 presents the mAP scores of the competing methods. The methods are grouped based on the dataset used during development. Our approach outperforms all methods in each group with a clear margin. The same result derived from the comparison of the PR curves is illustrated in Figure 6, with the light blue line (proposed approach) lying upon all others up to 90% recall in both cases. It is noteworthy that our approach trained on VCDB dataset outperforms four out of six methods, with two approaches achieving marginally better results, but both developed on the evaluation dataset.

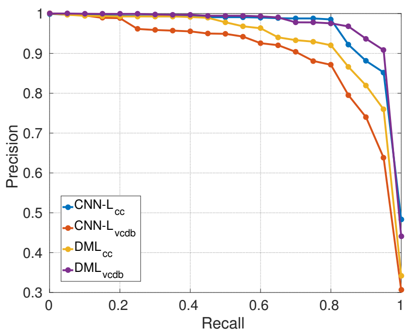
In our last experiment, we directly compared the old approach with the new one. For a fair comparison and to emulate more realistic settings, we build the old version with samples from VCDB. We then tested two variations, one that was developed on the CC_WEB_VIDEO dataset (same as D3.1) and another one developed on the VCDB dataset. For each of the 24 queries of CC_WEB_VIDEO, only the videos contained in its subset (the dataset is organized in 24 subsets, one per query) are considered as candidate and used for the calculation of retrieval performance. For a more challenging benchmark, we created CC_WEB_VIDEO* in the following way: for every query in CC_WEB_VIDEO, the set of candidate videos is the entire dataset instead of only the query subset (the videos from the other subsets are considered to be dissimilar).

Figure 7 depicts the PR curves of the four runs and the two setups. There is a clear difference between the performance of the two variants of the CNN-L approach, for both dataset setups. The proposed approach outperforms the CNN-L approach for all runs and setup at any recall point by a large margin. Similar conclusions can be drawn from the mAP scores of Table 4. The performance of CNN-L drops by more than 0.02 and 0.062 when it is trained on VCDB, for each setup respectively. Again, there is a considerable drop in performance in CC_WEB_VIDEO* setup for both approaches, with the proposed being more resilient to the setup change. The improvement of the new approach is considerable, reaching 4.0% at the CC_WEB_VIDEO* trained on VCDB data.

The new approach has approximately the same requirements in terms of processing time with the previous version. The feature extraction scheme is exactly the same, so the extraction time is similar



(a) CC_WEB_VIDEO



(b) CC_WEB_VIDEO*

Figure 7: Precision-Recall curve comparison of the proposed approach with two variants of the previous version of the method on two dataset setups.

	VCDB		CC_WEB_VIDEO	
	CC_WEB_VIDEO	CC_WEB_VIDEO*	CC_WEB_VIDEO	CC_WEB_VIDEO*
CNN-L	0.954	0.898	0.974	0.960
DML	**0.969**	**0.934**	**0.981**	**0.970**
improvement	1.6%	4.0%	0.7%	1.0%

Table 4: mAP comparison of the proposed approach with the previous version of the method on two different dataset setups and percentage of improvement.

to the extraction time reported in D3.1. For the calculation of the video embeddings, the processing time ranges between 1ms to 6ms depending on the utilized fusion scheme and CNN architecture, and it is almost the same as the indexing time of the old approach. The query time is the same for both fusion schemes and is according to the output vector length. It ranges from 20ms to 30ms per query. In conclusion, the improvement in terms of performance does not come at a cost in processing time.

In Figure 8, the results of two queries from our database are illustrated. Only a few selected videos are displayed for each example. In the first one, the candidate videos have been ranked successfully, with the near-duplicate videos at the top ranks and the irrelevant videos at the bottom. Moreover, the first irrelevant video appeared in the rank 82 with similarity less than 0.23. However, in the second example, irrelevant videos appeared also at top ranks and even higher than some near-duplicate videos. For instance, the second retrieved video in the example is an irrelevant video and has been ranked higher than the third which is a near-duplicate. Also, many irrelevant videos had similarity greater than 0.7 (second and forth videos). A probable explanation for this could be the fact that the query was a long video with many fast moving shots, captured vertically and with very poor lighting conditions. These factors could make it a challenging case and might cause many irrelevant videos to have high similarity. Analyzing the reasons why such false matches may appear is an important undertaking, and we intend to further study the issue with the aim of improving results. Nonetheless, we are confident that a learning system can be proven robust to such cases. The first solution to consider will be the training of our model with popular generalization techniques such as data augmentation.

With respect to user evaluations, test cycle 2 was run on the initial version of the service -the same that was presented in D3.1. The testers noted several bugs and issues with the service, both with respect to error handling (no checks for input URL format, empty parameter fields), and to processing bugs, such as not managing all input formats, or using ports that are often blocked by many corporate network connections. Furthermore, the status messages were noted to be not specific enough. In test cycle 3, errors were fixed and the feature extraction process was integrated internally of the service that considerably sped up indexing of new videos. Also, the delete call was added. Tester comments characterized the service as stable and reliable. Prior to test cycle 4, we implemented the update of the NDD method, the support of any video input and the addition of Near-Duplicate Localization. Tester comments were positive as well, reporting some bugs for not checking the type of input parameters (i.e. `async`, `force`, `t_sim`, `t_rank`).

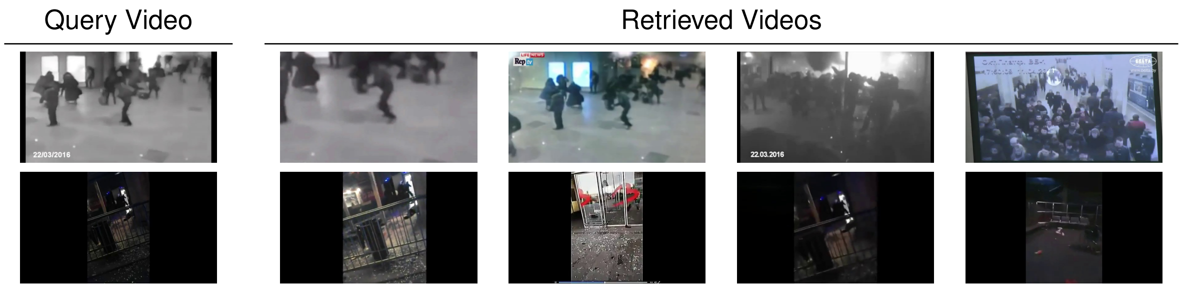


Figure 8: Selected videos ranked based on their similarity to the queries.

4.4 API layer and integration with InVID

During the second year of the project, the API layer was upgraded and its functionalities were extended. The service was updated with the implementation of the DML approach and support for a large number of different video sources was added, as well as better reporting about video information and the service status.

For the integration of the DML method to the InVID platform, we applied some modifications to the main approach. In Section 4.3, we empirically found that both fusion methods have almost the same performance. Hence, the videos in the database are indexed based on their individual frame embeddings. To speed up the retrieval process, an inverted file structure is build based on a codebook from frame vector sampled from VCDB dataset. Therefore, the similarity between two videos is computed as the cosine similarity of their aggregated representations.

In terms of the NDD service calls, several functionalities were added. The calls and their parameters are displayed in Table 5. Compared to the version reported in D3.1, a new delete call has been added that deletes a given video from the database of indexed videos. Two boolean parameters were added in the index call: a) `async` that makes the call asynchronous, and b) `force` that forces the service to re-index the given video, even if it is already indexed. Additionally, the search call no longer indexes the provided video; in case that a video does not exist in the index the respective error is returned. Two numeric parameters were added in the search call: a) `t_sim` that limits the returned videos whose similarity to the query surpass the given threshold, and b) `t_rank` that limits the returned videos up to those in the first ranks as determined by the given threshold. Furthermore, the NDD service supports the

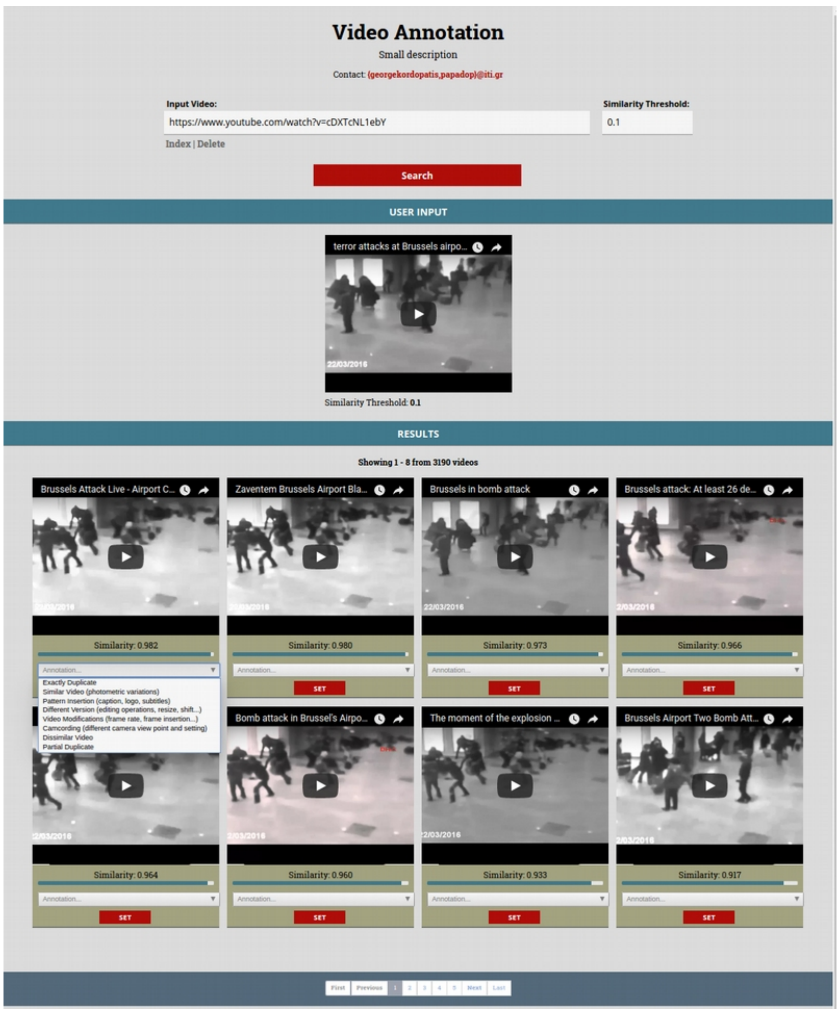
Table 5: Calls exposed by the Near-Duplicate Detection module API.

Service	**Request**	**URL**	**Parameter**
index video	GET	`/index`	`url=`<video url>
			`async=`<true or false>
			`force=`<true or false>
search video	GET	`/search`	`url=`<video url>
			`t_sim=`<similarity threshold>
			`t_rank=`<rank threshold>
delete video	DELETE	`/delete`	`url=`<video url>

indexing of video content from almost every popular video platform. We integrated internally the open source library `youtube-dl`[7] that allows to download videos from any given URL that contains videos. At the time of writing this deliverable, the video index powering the NDD search comprises more than 400,000 videos, approximately 300,000 of which were added as a result of indexing videos from past events, and 100,000 as a result of integrating the videos related to the trending topics detected by the WP2 service (which grow at a rate of 1,200 videos per day). Additionally, the service includes an early version of Near-Duplicate Localization that provides an indication of the sequences in a candidate video that are similar to a given query.

Additionally, we have developed an annotation tool (Figure 9) in order to construct a dataset for the comprehensive evaluation of the developed approached. The annotation tool provides all the functions supported by the NDD API, i.e. index, search, delete. Given a query video and a similarity threshold, it displays all the retrieved videos organized in pages based on their rank. The user can then annotate

[7]https://rg3.github.io/youtube-dl/


Video Annotation

Small description

Contact: (georgekordopatis,papadop)@iti.gr

Input Video: https://www.youtube.com/watch?v=cDXTcNL1ebY

Similarity Threshold: 0.1

Similarity Threshold: 0.1

Showing 1 - 8 from 3190 videos

Brussels Attack Live - Airport C...	Zaventem Brussels Airport Bla...	Brussels in bomb attack	Brussels attack: At least 26 de...
Similarity: 0.982	Similarity: 0.980	Similarity: 0.973	Similarity: 0.966
	Bomb attack in Brussel's Airpo...	The moment of the explosion ...	Brussels Airport Two Bomb Att...
Similarity: 0.964	Similarity: 0.960	Similarity: 0.933	Similarity: 0.917

Index | Delete
Search
USER INPUT
terror attacks at Brussels airpo...
RESULTS
Annotation...
Exactly Duplicate
Similar Video (photometric variations)
Pattern Insertion (caption, logo, subtitles)
Different Version (editing operations, resize, shift...)
Video Modifications (frame rate, frame insertion...)
Camcording (different camera view point and setting)
Dissimilar Video
Partial Duplicate
SET
First Previous 1 2 3 4 5 Next Last

Figure 9: Screenshot of the annotation tool.

the returned videos with respect to the query, essentially providing feedback on the algorithm and thus contributing to the creation of an annotated evaluation dataset. Besides being an independent feature, the annotation tool will also be used as base for the UI of the NDD module.

5 Logo Detection

Following the evaluation of the logo detection algorithm developed during the first year of InVID, and the identification of its most important shortcomings, during the second project year we decided to replace it with a different approach, designed to overcome those shortcomings. Using deep learning combined with an innovative training scheme that overcomes the problem of scarce training data, the new system is faster, more robust, capable of identifying semi-transparent or thinly shaped logos, and capable of reaching much greater accuracy than the one presented during the first year. Evaluations show its improvement with respect to speed, and our analysis highlights its potential for further improvement.

5.1 State of the art

In D3.1 we presented a survey of the state of the art, and the considerations that led to the design of the first version of our Logo Detection module. Specifically, we made the distinction between the typical use of the term "logo detection" in literature, which refers to finding instances of brand logos on displayed items (e.g. clothes) regardless of the viewing angle, and our task of identifying logos overlaid on videos, typically referred to as "TV logo detection"[8].

In principle, TV logo detection is a more constrained version of generic logo detection. However, in contrast to logo detection tasks which are typically tackled with machine learning over a small number of classes with a large number of training examples, the constraints of the TV logo task were exploited to build a simpler system, that could expand on more logo classes without requiring training. This was important since we cannot dedicate the manual effort of creating a large annotated training dataset, and even less so to collect a new training dataset every time we need to add a new logo to the system.

In D3.1 we had presented an algorithm based on keypoint extraction and matching. To learn a candidate logo, the algorithm simply extracted keypoints from a logo template. For detection in an unknown image, keypoints were extracted from the candidate image, then their descriptors were matched against the candidate logos, and RANSAC geometric modeling was run to keep only the keypoints that conformed to a consistent model. If a sufficient number of matching keypoints were found, a match was declared. Although this approach led to encouraging results during the first year of the project, it was reconsidered during the second year due to a number of issues:

- The time requirements for the keypoint-based approach scale linearly with the number of known logos as each candidate is evaluated separately. This would become a problem as the number of known logos increased, and low response times are a key requirement in InVID.
- As explained in D3.1, for each logo it was often necessary to include multiple variants, e.g. scaled, blurred, or pixelized. Thus the time requirements would increase even more steeply as more logos were added.
- The only way to successfully detect animated logos was to include instances of the logo in various stages of its animation in the template collection, further increasing the number of comparisons.
- Especially with respect to partially transparent logos, a large number of keypoints were found along the logo boundary. However, the local descriptor in that case was dependent on the background on which the logo appeared. Thus, there was often need to include logo templates both over dark and light backgrounds, multiplying the number of comparisons and increasing the detection time.
- Keypoint-based detection was very unsuitable for semi-transparent logos, which often vary radically depending on the background they are placed on.
- Despite the encouraging detection rates of the keypoint-based approach, there was little room for further improvement, as the method presented in D3.1 was optimized with respect to speed through parallelization, and with respect to performance using tricks such as extracting the mean image per shot.

Thus, the initial approach, while satisfactory as an initial solution, was hard to further improve in terms of accuracy, and was expected to become prohibitively slow with scale – even more so as, during

[8]In practice, we are not always dealing with formal TV channels and, most likely, the most important part of the task is identifying the logos of informal video sources such as paramilitary groups. However, we will be using the term "TV logo detection" as it is established in the literature.

the first test cycles, the response times using the initial logo dataset were already judged as too long by the users.

In reconsidering our choice of algorithm, we investigated a number of possible alternatives from the state of the art:

– One consideration was a sliding window approach (Chum & Zisserman, 2007; Ferrari, Fevrier, Jurie, & Schmid, 2008), where candidate overlapping windows would be extracted from the image, at multiple scales, a single descriptor extracted from each window, and compared to the descriptors of all candidate logos. While achieving high accuracy, such approaches are prohibitively slow for real-time evaluations and are becoming outdated.

– Another approach would be to use a region proposal algorithm to extract a small number of candidate regions from the image, and only evaluate these regions (Gu, Lim, Arbeláez, & Malik, 2009). While faster than sliding window methods, these approaches also require several seconds to propose the candidate windows, and preliminary experiments showed that in many cases none of the proposed regions contained the logo.

– The best performance in object detection is currently achieved using Deep Neural Networks, and specifically Region proposal Convolutional Neural Networks (R-CNN) (Girshick, 2015; Ren, He, Girshick, & Sun, 2015a). These methods train a region proposal network together with a classification network, and are very fast at detection time since they only require a single forward pass to return both classification and localization information. However they typically require a lot of training data –which, as stated above, we could not provide.

5.2 Method description

Based on its merits, we based our Logo Detection module on the third of the above approaches, and devised a number of solutions to overcome the lack of training data and make the service scalable, fast, and at least as accurate as the previous implementation.

As network architecture, we have chosen the Faster Region-proposal Convolutional Neural Network (Faster-RCNN) (Ren, He, Girshick, & Sun, 2015b). This architecture simultaneously outputs a large number of region proposals and classification estimates for each region in a single pass, making it extremely fast during detection. Furthermore, the region proposal and the classification parts are trained simultaneously, making its training faster than its counterparts. Its performance is among the best in the state-of-the-art, and open-source implementations exist for the Caffe[9] and Tensorflow[10] frameworks. Thus, it is easy to experiment and adapt to the project's needs.

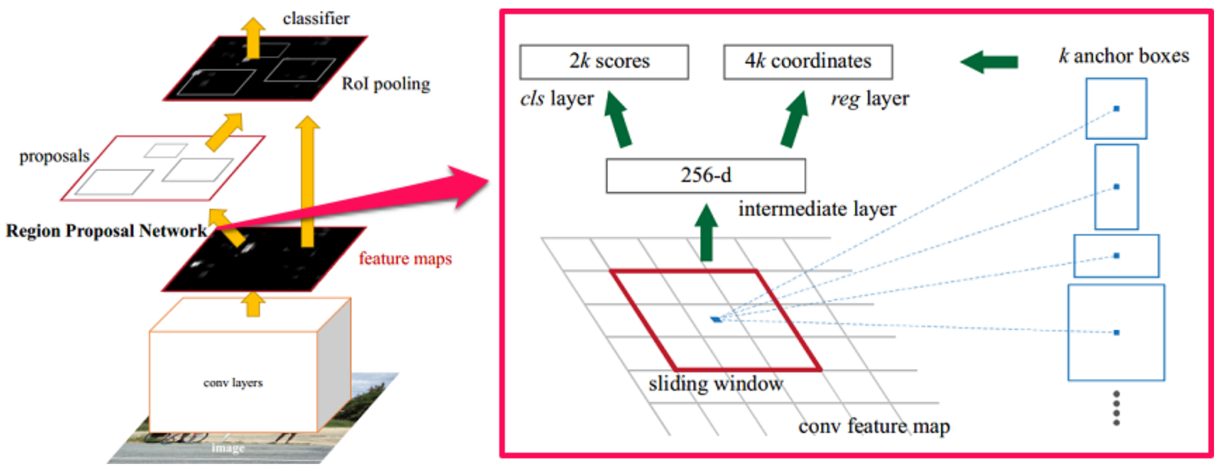


Figure 10: The Faster-RCNN architecture (image taken from (Ren et al., 2015b)).

The major issue with Deep Neural Networks is training. They tend to require large amounts of annotated data and generally require a lot of time to train. However, the task at hand is significantly simpler than most detection tasks, since in our case the candidate object (i.e. a logo) has very little

[9]http://caffe.berkeleyvision.org/
[10]https://www.tensorflow.org/

variability between instances. This characteristic allowed us to use an innovative training technique that removes the need for manually annotated data.

Normally for an object detection task, we would require a large number of annotated images (hundreds or thousands) containing each object, and each image would have to be annotated with the class and the localization coordinates of the object. Creating such a training dataset for logos would be impossible within the scope of InVID, and impractical when extending the dataset with new logos. However, since in our case all appearances of a logo would be expected to be very similar, we were able to devise a way to automatically generate training images on-the-fly using a single logo example. A training image can be generated by taking a random base image from any realistic image dataset (such as MIRFlickr[11]), and a logo from our collection, and placing the logo at a random position on the image. To account for variations of the logo, a set of data augmentation techniques are applied, such as scaling (sometimes non-proportional), blurring by a random-sized kernel, brightness and color modification. In this way, we can generate a practically infinite number of training images. To further speed up the training process, we place a number of logos in each training image, in a non-overlapping manner, ranging from 1 to 3 (Figure 11). This process allows us to train a classifier without using a manually annotated dataset. It also allows for extensibility, since adding a new entry in the list of known logos does not require additional examples, but only a single logo template. Following training, the detection process is simple: an image is passed through the trained model, and the model outputs a list of region estimates, plus the estimate of the logo class that was detected within them.

Furthermore, the new, CNN-based approach has no limitations with respect to the logo background and potential transparency, provided it could be trained with enough representative examples.

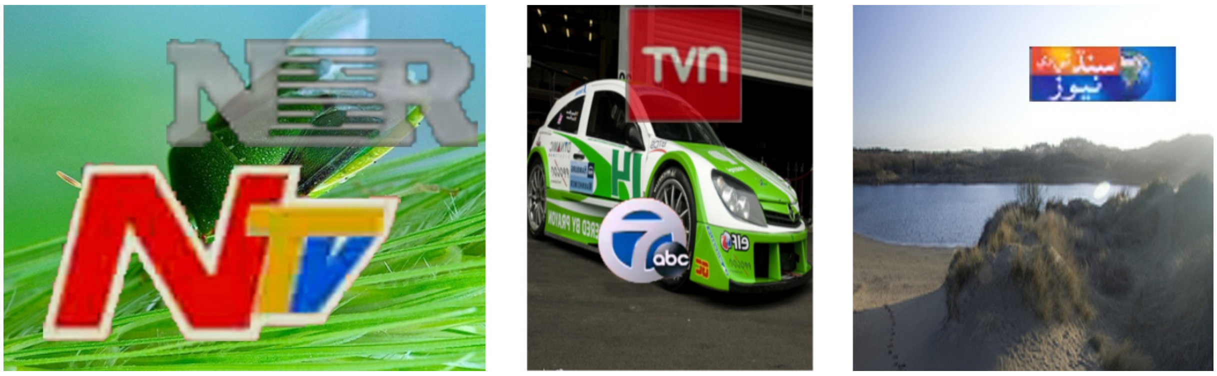


Figure 11: Three artificially generated training samples with logos, some of which are semi-transparent.

5.3 Evaluation and progress since Year 1

Having designed the algorithm, we proceeded to implement it using an existing framework, namely the `py-faster-rcnn`[12] implementation for Caffe. As `py-faster-rcnn` is designed to take annotated training datasets, we converted the code to operate with images generated on-the-fly. During our evaluations, this was necessary as we did not know beforehand the number of training images that would be needed for training. Having established that, it would be simple to automatically generate the entire training dataset prior to the training process and greatly speed it up.

The model was trained on an PC equipped with a Maxwell Titan X GPU. We used the VGG16 architecture (Simonyan & Zisserman, 2014) for the convolutional part of the network. For training, we used transfer learning and initialized the weights using a network pre-trained on the ImageNed classification task (Deng et al., 2009b). Currently, the training process converges after roughly 48 hours. Given any image, the trained model returns a number of overlapping regions (determined by their top left and lower right corner) and for each region, one value per logo class indicating the probability of the region containing the corresponding logo. Given the specificity of the task, most logos are detected with extremely high certainty, so we threshold the probabilities at 0.99. For localization, we return the alpha-trimmed mean of all region boundary coordinates containing the same logo, which - following qualitative evaluations - gives a good estimation of the logo location.

[11] `http://press.liacs.nl/mirflickr/`
[12] `https://github.com/rbgirshick/py-faster-rcnn`

In order to evaluate our improvements compared to the first year of the project, we ran the same evaluations as in D3.1, using the same dataset and logo classes. The comparative results are presented in Table 6

Table 6: Logo detection evaluation results

	Videos			Shots		
	Keypoints	Fr-RCNN 1	Fr-RCNN 2	Keypoints	Fr-RCNN 1	Fr-RCNN 2
True Detections	0.83	0.80	0.85	0.63	0.64	0.72
False Detections	0.06	0.06	0.13	0.004	0.01	0.01

As shown in Table 6, the Faster-RCNN version of the algorithm is currently comparable to the keypoint-based approach. We tested two RCNN models, one trained with early stopping (Fr-RCNN 1) and one trained for longer period of time (Fr-RCNN 2). Fr-RCNN 1 shows slightly lower True Detection (TD) rates than keypoint-based methods, and comparable False Detections (FD). On the other hand, Fr-RCNN 2 has better TD rates, but significantly higher FD rates. One explanation that the logo template collection contains several images of relatively low quality that are blurred. For the keypoint-based method these were necessary, in order to be able to detect logos in low-quality images. However, in Faster-RCNN training, especially after the potential additional blurring of the augmentation step, the network might be trained on extremely blurred templates, which would lead then to finding false matches on non-relevant regions. Another observation is that the false positives appear disproportionally higher per video than per shot. This means that the relatively few false positives in the shots (0.01) are very scattered across the shots, with few (usually one at most) in each video. Thus in practice these spurious matches are not distracting for professionals, since they are easily discarded by visual inspection.

Overall, however, we consider the Faster-RCNN approach to be a superior choice, for two reasons: 1) the results for Faster-RCNN have significant potential for improvement by improving the template dataset – with the help of the user partners – and by tweaking the training parameters, and 2) the Faster-RCNN approach is significantly faster, and its detection speed is much less dependent on the number of recognizable logos. To confirm the new methods superiority, we ran a series of evaluations with respect to detection speed. For fairness, we had to account for certain additional computational costs that the Faster-RCNN algorithm requires. Specifically, as the neural network runs on a PC equipped with a GPU, it had to be placed on a separate server, and it is possible that the communication between the logo detection server and the neural network server may incur additional delays. So, the comparison was run between two services. This means that the reported times include the service communication delays, which reflects the actual user experience. Table 7 gives the current differences in speed between the two services, per single image, per video shot, and per video. The reasons that the performance per shot is improved more than the performance per image, is that a) the keypoint-based method was run on both the middle image and the mean image of the shot in order to reach its optimal performance, while the Faster-RCNN algorithm only runs on the middle image of each shot and b) the impact of the communication overhead is much smaller, since the major load is accessing the image/video, which only happens once per video. In fact, the speed of the new service is so superior that it outweighs even the added time requirements of fragmenting the video (which we do not have in images), leading to the much higher per-shot improvement compared to the per-image one.

Table 7: Logo detection time requirements (in seconds)

	Image	**Shot**	**Video**
Keypoint-based	8.47	6.56	383.50
Faster-RCNN	4.17	1.18	69.00
Speedup	203%	556%	556%

While it is conceivable that adding many new logos may increase the training time, since the time is relatively low, we consider that any potential increase will be manageable. Furthermore, it is possible that the overall training time can be reduced by tweaking the training hyperparameters and improving the data augmentation procedure.

With respect to user evaluations, the first test cycle was run on the initial version of the service -the same that was presented in D3.1. The testers noted several bugs and issues with the service, both with respect to error handling (no checks for input URL format, empty parameter fields), and to processing bugs, such as not managing all input formats, or using ports that are often blocked by many corporate network connections. Furthermore, the error messages were noted to be not specific enough.

In test cycle 2 all these errors were fixed. Furthermore, the service was sped up through parallelization -as an attempt to make the keypoint-based fast and scalable. While several more error checks were found to be missing, generally the testers noted the increased speed and reliability. At that stage, while the testers were satisfied with the service speed, we nevertheless began designing the new, CNN-based service, knowing that the keypoint-based approach would not be able to scale to many more candidate logos. Other comments included the testers' inability to find the UI option for choosing between a video submission and an image submission, and the absence of a provided list of known logos to allow testers to know which logos should be detectable by the service.

In test cycle 3, these errors were fixed and a timeline-based output format was added as an option. Also, the option to return a list of known logos was added, as well as automatic detection between videos and images. Tester comments characterized the service as stable and reliable, and all focus was on various false positives and false negatives returned by the service. Overall, while detection performance has increased significantly and the new algorithm can identify logos which would have been very hard for the previous one, some false detections still remain, as shown in the last example of Figure 12.

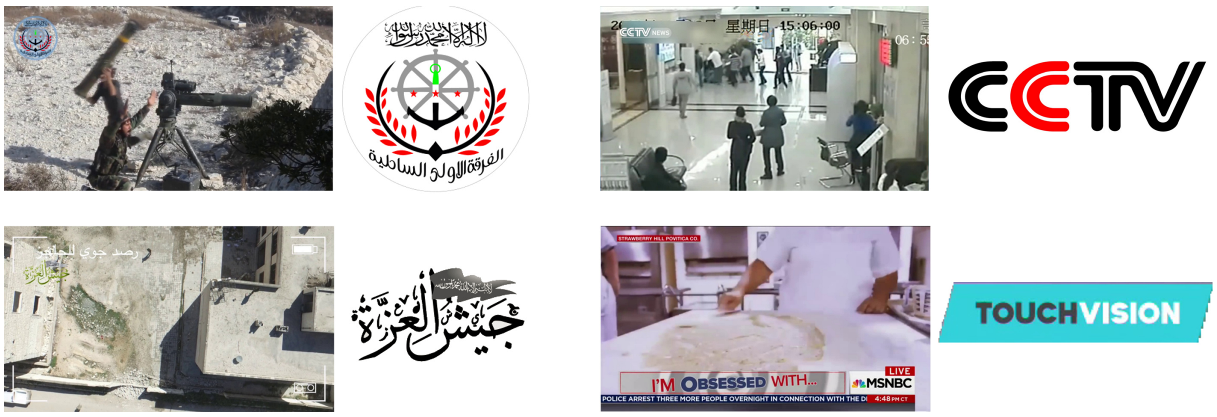


Figure 12: Three examples of successful logo detections and one example of a false one.

5.4 API layer and integration with InVID

During the second year of the project, the API layer was extended with more functionalities. Support for more video sources was added, as well as new API calls to provide additional information on the service and its status.

The video source coverage was extended to include Facebook, Twitter, and Dropbox videos, as well as video files directly uploaded on HTTP servers. This coverage essentially matches that of the Video Fragmentation and Annotation service from WP2. Since the fragmentation and keyframe extraction takes place there, the Logo Detection service simply verifies that the submitted URL is indeed within the spectrum of covered sources, and then submits it for fragmentation. Another modification is that the service now uses the sub-shot fragmentation option, since it takes much less time to fragment the video. Given that the detection step takes very little time with the Faster-RCNN algorithm, even with the increased number of keyframes the overall times are still greatly reduced.

Extensions to the API include: a) an automatic content type detection GET request that takes a URL and returns the item type ("image" or "video"); b) a GET request that returns the analysis status of an item in the dataset; c) a GET request that returns the list of known logos and the associated templates and Wikipedia links; and d) a GET request to submit a new logo template to the service alongside its name and associated Wikipedia link. The updated list of calls exposed by the Logo Detection module API is presented in Table 8.

The automatic content detection can be used by a UI to automatically redirect calls to the appropriate service (image or video) without waiting for an explicit choice by the user. The analysis status is useful to get the results when uploading a file using a POST request. The list of recognizable logos is drawn directly from the model, thus it does not require separate updates when expanding the logo list. The feature was deemed necessary during the test cycles, since testers often tried to test the service with

Table 8: Calls exposed by the Logo Detection module API. New calls are marked in bold.

Service	URL	Parameter	Notes
image from URL	`/fromimageurl`	`url=`<image url> `timeline=`<1 or 0>	
video from URL	`/fromvideourl`	`url=`<video url> `timeline=`<1 or 0>	YouTube, DailyMotion, Facebook, Twitter, Dropbox, or video file.
image from InVID	`/fromimageid`	`id=`<InVID image id>	
video from InVID	`/fromvideoid`	`id=`<InVID video id>	
image from file	`/fromimagefile`	`imagefile=`<the image file>	POST request
identify content	`/fromurl`	`url=`<content url>	
analysis status	`/analysisstatus`	`id=`<item id>, `timeline=`<1 or 0>	Useful for tracking “image from file”
get logo list	`/logolist`		
submit new logo	`/submitlogo`	`imageurl=`<logo image URL> `logoname=`<logo class name> `wiki=`<Wikipedia link>	

logos that were not in the list. Finally, the submission of new logos by the users ensures that the service will remain relevant in the future, by allowing users to extend it. Since CNN training is an offline process, the service keeps the items in the server alongside a database entry with the name and Wikipedia URL, pending inspection and processing by the service administrators. In order to be added to the dataset, the logos have to be inspected, cleaned of any background, and added to the list of training templates. Then in the next training cycle (which can take place at regular intervals), they will be incorporated in the model.

Furthermore, following an update to the Verification App requirements, an option was added to the analysis calls (`/fromimageurl`, `/fromvideourl`, `/analysisstatus`) that modifies the ouput format. While in the original implementation, the JSON output contained one entry per shot, with all the logos detected in it, the new, timeline-based output format contains one entry per detected logo, giving all the time intervals during which the logo appears in the video. Figure 13 shows a comparison of the old output format and the new, timeline-based one.

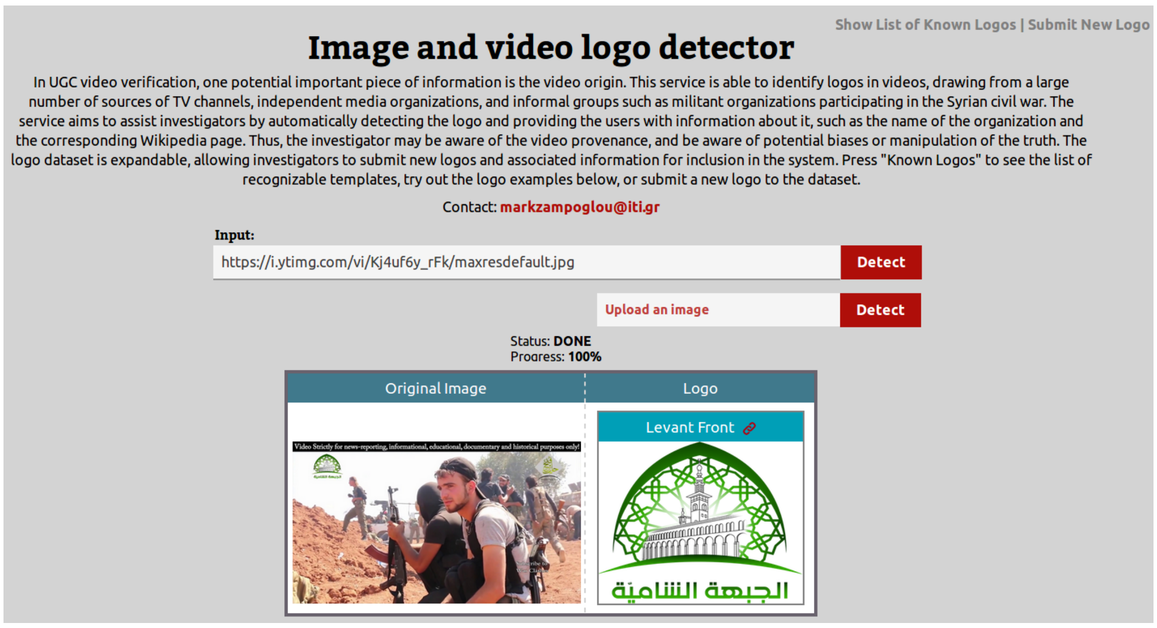
In parallel to the API and integration with the Verification App, we have also developed a demo UI hosted in our service, for testing and demonstration reasons. The UI offers a short description, instructions on how to use the service, and several image and video examples of successful detections. The demo UI works with all video sharing platforms supported by the service (YouTube, DailyMotion, Facebook, Twitter, Dropbox) and furthermore is able to scan links and automatically detect whether we are dealing with a video or image item, and use the appropriate service call.

It also offers the option to submit any type of recognizable object (images and all accepted videos), essentially providing a front-end to the logo submission service, where the user can submit a logo image alongside a name and a Wiki URL, and the data are stored in the service for future integration by us. Finally, the UI provides access to the logo list, returning a visualization of all logo templates currently recognizable by the system. A sample detection visualization can be seen in Figure 14. It shows the URL of a photograph uploaded by a Syrian rebel group, and the results of the detection: the name of the group and the logo template that was matched to the one in the image. The name of the group (“Levant Front”) redirects to the corresponding Wikipedia page, while the two links that can be seen at the top right corner redirect the user to the list of known logos and to the new logo submission form respectively.

```
{ "status": "DONE",
  "url": "https://www.youtube.com/watch?v=wE5qBDYeo7g",
  "progress": 100,
  "detectionResult": [ {
      "sampleKeyframe": "http://logos.iti.gr/...",
      "shotID": 1,
      "detectedLogos": [ {
          "wikiURL": "https://en.wikipedia.org/wiki/BBC",
          "frameLocalization": "[290.1, 11.5, 380.8, 59.1]",
          "logoName": "BBC",
          "logoURL": "http://.../BBC-Logo.png"
        } ] },
    {
    ...
    {
      "sampleKeyframe": "http://logos.iti.gr/...",
      "shotID": 16,
      "detectedLogos": [ {
          "wikiURL": "https://en.wikipedia.org/wiki/BBC",
          "frameLocalization": "[293.4, 11.9, 381.9, 62.5]",
          "logoName": "BBC",
          "logoURL": "http://.../BBC-Logo.png"
        } ] } ],
  "message": "",
  "_id": "wE5qBDYeo7g",
  "type": "urlvideo" }
```

```
{ "status": "DONE",
  "url": "https://www.youtube.com/watch?v=wE5qBDYeo7g",
  "progress": 100,
  "detectionResult": [ {
      "wikiURL": "https://.../BBC",
      "begin": [
        "0.040",
        "3.160",
        ...
        "60.760",
        "63.400" ],
      "end": [
        "3.120",
        "4.560",
        ...
        "63.360",
        "66.240" ],
      "logoName": "BBC",
      "logoURL": "http://.../BBC-Logo.png",
      "frameLocalization": [
        "[290, 11, 380, 59]",
        "[287, 12, 381, 60]",
        ...
        "[289, 12, 380, 61]",
        "[293, 11, 381, 62]" ] } ],
  "message": "",
  "_id": "wE5qBDYeo7g_timeline",
  "type": "urlvideo" }
```

Figure 13: Left) Earlier shot-based output format, Right) New timeline-based output format.



Show List of Known Logos | Submit New Logo

Image and video logo detector

In UGC video verification, one potential important piece of information is the video origin. This service is able to identify logos in videos, drawing from a large number of sources of TV channels, independent media organizations, and informal groups such as militant organizations participating in the Syrian civil war. The service aims to assist investigators by automatically detecting the logo and providing the users with information about it, such as the name of the organization and the corresponding Wikipedia page. Thus, the investigator may be aware of the video provenance, and be aware of potential biases or manipulation of the truth. The logo dataset is expandable, allowing investigators to submit new logos and associated information for inclusion in the system. Press "Known Logos" to see the list of recognizable templates, try out the logo examples below, or submit a new logo to the dataset.

Contact: markzampoglou@iti.gr

Input: https://i.ytimg.com/vi/Kj4uf6y_rFk/maxresdefault.jpg Detect

Upload an image Detect

Status: DONE
Progress: 100%

Original Image	Logo
	Levant Front

Figure 14: A screenshot from the logo detection demo UI.

6 Location detection

During the second year of the project, we significantly improved two major components of the location detection approach: popularity and context analysis. We further developed a novel geolocation dataset for evaluation, and ran extensive evaluations on this plus two other datasets. In this section we present our improvements on the Location Detection algorithm, we describe the new, StoryLens corpora dataset and we demonstrate the increased performance of the new algorithm and its superiority with respect to other related tools.

6.1 State of the art

Named Entity Linking (Ji et al., 2016) is becoming more important as more and more entities are present on the web with their official URLs or DBpedia URIs (Lehmann et al., 2015) or Wikidata (Vrandecic & Krötzsch, 2014). All the main classes of entities (Person - PER, Organisation - ORG, LOC - Location) that were included in the classic formulation of the NEL problem are difficult, but geolocation is by far the most difficult class as it has a lot of conflicts with all the other classes (e.g. street names often have people names, there are many confusions between organizations and the buildings in which they are located). When expanding the number of classes by splitting Location into multiple classes based on the type of location entity or simply adding new classes (e.g. Event - EVE/EVENT, Miscellaneous - MISC, Product - PROD) these conflicts compound. The expansion of the Location entity type into at least three types is however necessary in order to move the state of the art further, and in the last challenges has become the standard. The location types proposed by NIST for the TAC KBP challenge (Ji et al., 2016) included the following: **Natural Locations** like mountains, rivers designated by the abbreviation **LOC**, **Geo-Political Entities** like countries, regions, cities, streets designated by the abbreviation GPE and **Facilities** like airports, road infrastructure, parks or buildings designated by **FAC**. While many systems still use the old classification with three classes (only a single location type), especially the system that still rely on the Stanford tagger with three classes, it is important to move towards expanded classifications in order to improve the results.

Due to business needs, geoparsing has always been the most important component of NEL systems. Initial efforts were focused on parsing GPS data and in time the semantics components were added, the focus being shifted towards geosemantics. Current NEL systems provide geosemantics functionality out-of-the-box since the disambiguation algorithms need to work regardless of the entity type. The most successful classes of NEL systems belong to the following three categories: a) graph-based disambiguation - AIDA (Hoffart et al., 2011) and AGDISTIS (Usbeck et al., 2014); b) mixtures of Conditional Random Fields models - ADEL (Plu, Rizzo, & Troncy, 2016) or c) neural models - Lample's system (Lample, Ballesteros, Subramanian, Kawakami, & Dyer, 2016). Our own tool, Recognyze, can be included into the graph-based disambiguation tools. Each year, several challenges are organized, the most important being the NIST's TAC-KBP (Ji et al., 2016). During the last year the competition was expanded from 3 to 13 languages, pilots for languages of some smaller countries (e.g. Albania) being launched. While we have not participated to this challenge, our tool already provides support for multiple languages (e.g. English, German, French, Spanish, Italian, Czech), some of these languages being integrated into the InVID project.

The NEL field is going through a period of consolidation, with two significant surveys appearing in the last year alone. (Rizzo, Pereira, Varga, van Erp, & Basave, 2017) summarizes the lessons learned during the several editions of the NEEL Challenge with a focus on changes to the annotation methodology, corpus analysis, emerging trends in the design and evaluation of NEL system. The work also includes a long analysis of the evaluation measures (e.g. scorers) used during these challenges and is notable for the inclusion of all the major systems that were launched in the last five years. (Ozdikis, Oguztüzün, & Karagoz, 2017) is focused strictly on location detection techniques from short texts (e.g. Twitter) and analyzes the best algorithms for jointly estimating the real location of Twitter events.

As it can be seen from (Ozdikis et al., 2017), in some cases, instead of fine-tuning general-purpose NEL tools, researchers have chosen to create new tools focused only on geosemantics. The scalable architecture used for geoparsing and geosemantics extraction in the REVEAL project (Middleton & Krivcovs, 2016) included features like the tweet content, position of the terms, part-of-speech (POS) sets, and 3-gram feature sets that combined named entities with their POS tags. A different example of geosemantics tool was the one develop by Hanan Samet's group (Samet et al., 2014) which focused on map-based disambiguation algorithms. The idea behind it was that entities from a document need to be disambiguated directly on map with a bounding box, therefore limited to a certain area. The common criticism to this approach is generally the fact that the granularity of the map is not always dynamically chosen (e.g. a mention of Paris in a set of documents that refer mostly to the U.S. might lead to the U.S. city instead of the French capital). In our tool we have tried to combine some of these approaches, essentially creating a map-based disambiguation on top of the graph-based disambiguation algorithms.

6.2 Method description

Recognyze is the named entity recognition and linking (NER/NEL) component integrated into the InVID Platform. The current generation of Recognyze uses graph-disambiguation algorithms. The main idea behind this class of algorithms is using the links between the entities extracted from a text in order to perform the correct disambiguation and linking to Knowledge Bases (KB). In the past year in InVID we have concentrated on the improvement of NER/NEL coverage and accuracy when dealing with references to geographical locations, especially within shorter texts (tweets).

The Recognyze framework contains the following components:

– Linked Data Sources - We have used various versions of some widely used Knowledge Bases (KBs) like DBpedia, Wikidata, Geonames exposed as TDB datasets for a Fuseki triplestore. We have chosen TDB due to its scalability and Fuseki due to the federation capabilities, as often our queries involved combining attributes from multiple KBs.

– Analyzers - A large collection of analyzers (e.g. some widely used Name Analyzers include Capitalization or Abbreviation Analyzers) can be used for creating different types of heuristics for disambiguating Named Entities.

– Dictionaries - A collection of terms (e.g. affixes, rare words) that is used for blacklisting or for identifying entities of different types.

– Filters - A set of classes that contains the business logic for filtering entities or relations.

– Disambiguation Algorithms - A set of algorithms (e.g. Unique Candidate, Multiple Knowledge Bases) that is used for disambiguating the candidate mentions.

– Lexicons and Profiles - The entities (lexicon) and the settings (profile) needed for performing a disambiguation.

– JAIRO - An external enrichment library that formats Recognyze input.

– Recognyze API - The private API available through a Swagger interface.

– Recognyze Clients - These are Java or Python clients that provide wrappers for the most important methods from the Recognyze API.

Named Entity Linking solutions are typically expensive and they require a lot of RAM (e.g. DBpedia Spotlight requires typically around 100 GB of RAM, AIDA close to 1 TB). Recognyze requires between 50 and 400 GB of RAM, depending on how it is deployed (e.g. if all the different profiles are deployed or only specific ones, if all KBs are needed).

The quality of the results is dependent on all the components working well, but nevertheless, it is clear that the quality of the sources, the filters used and the disambiguation algorithms have the greatest impact on the quality of the results.

In order to provide good results we have only ingested cleaned versions of the KBs in our Fuseki TDB instance. We have not removed bad results, but we made sure that the KBs do not have broken links, wrong formatting or results in other languages. Where possible, the KBs were interlinked (e.g. by creating new links based on the Wikipedia links) in order to ease the SPARQL federation queries.

Filters are important in the sense that graph-based disambiguation algorithms only perform well with clean results and quality links between the entities. Our algorithms can be seen as a natural extensions of those proposed by AIDA and Hasan Samet. In graph-based disambiguation, the result of the disambiguation has always been chosen based on a certain metric. Some popular metrics have included the assumption of uniqueness, popularity or context. As far as the assumption of uniqueness goes, this can only be applied to a small set of entities that are truly unique, but the rest of the entities will have to be disambiguated using other assumptions. Popularity can be modelled in different ways in graphs, depending on the goal, therefore the most well-known metrics for measuring popularity have been centrality, number of links, or PageRank. Regardless of which popularity measure is used, the graph will tend to be biased towards popular entities. In some cases only additional details or context (e.g. additional data from the abstract or data that only makes sense for certain types of entities) will help settle the real entity.

Our initial implementation contained basic versions of the uniqueness, popularity and context algorithms, but we have subsequently fine-tuned them. While we have not changed the uniqueness algorithm, the other two have been changed significantly during the last year. Initially we only focused on the number of links as a popularity measure, but now we have switched to a PageRank based-measure. The main change was to cut all the links that could potentially be shared by all entities from the graph associated to a text (e.g. links to Wikipedia or to the Thing type) before performing the PageRank computation. The reason for removing these links was the fact that they created fake connections between entities without any real support and hardened the disambiguation proccess. The context information was improved by adding several attributes that are specific to location entities (e.g. geographic coordinates, country). Our algorithm is somewhat similar to Hasan Samet's map-based disambiguation, but the bounding box is not given through precise geographical coordinates, but rather through geographical hierarchies. For example, in a text related to president Bush and Texas, a mention to Paris will surface Paris, Texas, except for the situation in which multiple other entities from a superior geographical hierarchy are mentioned (e.g. Berlin, Washington, London), in which case Paris, France will be returned.

6.3 Evaluation and progress since Year 1

The evaluation was focused on the following major areas: i) creating a reusable geolocation dataset focused on different types of textual content (e.g. tweets, subtitles, news articles); ii) evaluating the performance on multiple datasets.

6.3.1 The StoryLens corpora

StoryLens [13], the corpora described in this section, was initially created in order to improve the geolocation functionality of our tool (Recognyze). Later it was expanded in order to cover events and other entity types as well. It is publicly available on GitHub.

Context should be key when consuming news, regardless of the source, but news media monitoring is full of noise due to massive retweets, rethreads, biases or fake news. Multiple similar events are often grouped based on the subject, entities, time and events into single narratives, becoming episodes of story arcs, while singular events are often neglected if they are not related to a big tragedy. When news media outlets connect unrelated events in order to craft better narratives, the automated tools used to collect, analyze and visualize data about current news events (e.g. detecting breaking news from social media, crawling online media about news events) follow suit. While the early tweets related to a news event might all be from a particular location, once the event is broadcast news items related to it will quickly accumulate geographical mentions from all over the world due to people's reactions, making it difficult to rely on automatically generated geolocation metadata in order to find eyewitnesses or real footage from the scene. Complicating the matter even further, corporation biases or the rise of fake news show that even big media outlets cannot always be relied upon to deliver straight facts. We think that in order to correctly deconstruct the media landscape it is best to apply different lenses when analyzing the data. One method of applying different lenses for such news media monitoring systems can be to apply different types of annotations to the same data, for example annotations that take into account entity types, overlaps, stories or even the differences in style and content between media sources (e.g. that tweets are shorter and full of abbreviations compared to longer textual news articles). We call a corpus created through such a method a multiview (or multiple lenses) corpora, as we consider its creation process to be somewhat similar to the one used by photographers when trying lenses with various ranges or focal distances in order to find the right one for shooting a certain scene. In this section we present the process through which such a corpus called StoryLens was created, as well as the various tasks and evaluations that can be later performed in order to improve the automated news media monitoring processes.

In order to build a multiple lenses corpus, a new Python framework was designed. It contains three components: i) Annotations - for documents selection, annotation extraction, links mining or clustering; ii) Lenses - focused on creating new types of lenses (e.g. links between the same entities or between events and stories); iii) Evaluation - for providing statistics on the content of the corpus, as well as on the performance of various NEL tools on this corpus.

[13] https://github.com/modultechnology/storylens

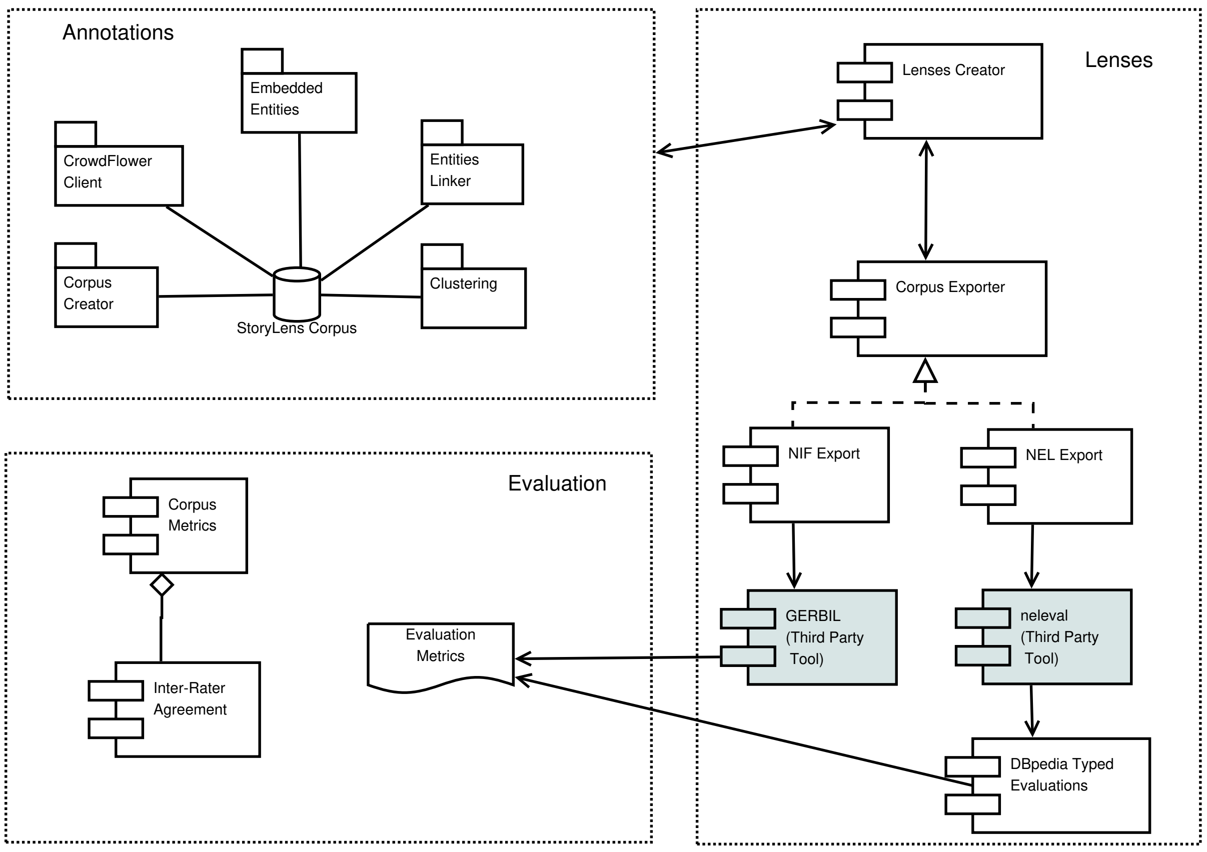


Figure 15: Framework used for corpus construction.

Corpus Creation and Annotation Process. During the process of differentiating real from fake news, we have collected a large number of documents from different types of media, including (but not limited to) news media and social media (e.g. tweets, YouTube subtitles) with the stated purpose of evaluating geolocation detection. A package to select some documents in order to evaluate our annotations was soon created. The extracted documents were split into three partitions based on the content's provenance: i) news articles, ii) subtitles, iii) tweets.

The initial plan was to simply evaluate the three location types included in the TAC-KBP (Ji et al., 2016) challenges: GPE - Geo-Political Entities (e.g. countries, regions, cities), LOC - Natural Locations (e.g. mountains, rivers, lakes) and FAC - Facilities (e.g. buildings, infrastructure). Because of the serialized nature of news media, there was a need to correctly repair the various errors caused by the location entities embedded in the other entities (e.g. *Grenfell Tower Fire* event title embeds the *Grenfell Tower* FAC entity). As a first step we decided to expand the number of entity types included in the corpus and also added PER - Person, ORG - Organisation, EVENT - Event. In order to make the annotations work on several levels (e.g. event or story levels) we decided to add new annotation sets in order to account for the correct nested entities (also known as embedded entities), differentiate between events or stories, or understand the links between entities, therefore building multiple views over the same texts. The annotation rules were included in an Annotation Guideline that was similar to the ones used for TAC-KBP and related semantic evaluation challenges. The ontology described in the guideline contains the following classes: PER - Person, ORG - Organisation, GPE - Geo Political Entity, LOC - Natural Location, FAC - Facility, EVENT - Event, WORK - Work of Art, PROD - Product and MISC - any other type. For each class we provide a set of examples and a set of rules in order to guide the annotators.

The annotators were asked to completely disregard any embedded (nested) entities. The embedded entities were later added automatically in a separate lens. Each document was annotated at least twice. An annotation was provided either by a person from our group or by a random person using the CrowdFlower API[14] in order to have a somewhat balanced view over what people considered correct entities. Our assumption was that people trained in-house will probably be more biased towards the

[14] https://www.crowdflower.com/

Category	Stories
Politics	Obamacare Repeal, Philando Castile, Brexit, London Attack (May 2017)
Entertainment	Bill Cosby Scandal, Coachella, Wonder Woman
Disaster	Grenfell Tower Fire, USS Fitzgerald Collision, Orlando Shooting, WannaCry
Business	Amazon Buys Whole Foods, ExxonMobil XTO Denver Office Closing
Sports	UEFA Champions League Final, Europa League Final, Roland Garros, Wimbledon

Table 9: Example stories included in the corpus.

Category	Description	Example
Gold	The entities from the gold	N.Y. Giants Stadium
Long	Longest match without overlaps	N.Y. Giants Stadium
Embedded	Entities including overlaps	New York, N.Y. Giants Stadium
DBpedia Version	Different DBpedia Version	MetLife Stadium

Table 10: Examples of lenses included in the dataset.

traditional view on named entities and more predisposed to follow the guidelines, whereas the remote annotators that used CrowdFlower or were new to the subject would not have such bias. All documents were annotated at least twice and the results were judged by the corpus creator. The annotators were asked to provide information about the surface forms and entity types. The online Wikipedia version was used for providing the entity links in order to ease the task. The Wikipedia links were later transformed into DBpedia links via SPARQL queries. Where such links were missing, we have constructed them, but added an additional comment in order to explain this provenance. The unlinked entities (usually called NIL) were grouped together using the Hierarchical Clustering algorithm from the scikit-learn Python package[15].

Lenses Creation. In order to define a new lens it is generally required to implement a new package that defines its functionality. Several lenses are described in the next subsection. The output of the various lenses is typically exported to the NIF (Hellmann, Lehmann, Auer, & Brümmer, 2013), CSV or TAC-KBP (Ji et al., 2016) formats. Since NIF does not support multiple annotation sets, we generally provide different NIF outputs for the various sets (e.g. with or without embedded entities). For typed evaluations the output only contains the matching entity types (e.g. for geolocation only GPE, FAC and LOC types are taken into account), whereas for TAC-KBP evaluations it is restricted to the five classic types that are typically required (e.g. PER, ORG, GPE, LOC, FAC). A different output file contains all the higher level annotations (e.g. stories). The output for the Twitter partition of the corpora only contains the annotations due to copyright restrictions, but the actual texts of the tweets can be downloaded by IDs using free scripts[16].

We have created several lenses in order to explore the texts contained in the corpus, as it can be seen in Table 10. The classic lens is called *Long* as it does not include any overlaps between the various entities (e.g. *Trump Hotel DC* would not be annotated two times to account for all the possible cases: *Trump*-PER, *Trump Hotel DC*-ORG), only the longest match for a surface form being considered. The lens that includes the overlaps is called *Embedded*. The *Stories* lens includes the set of documents related to a particular story as we also wanted to capture some information related to long-running narratives. The *Events* lens expands upon the previous lens and collects all the events related to a particular story, offering us some insights into the development of a story when the various texts are arranged chronologically. The *Links* lens collects all the links between the entities that appear together in a single text. A simplified version of it includes only the counts of relations between each two entities as represented by their links. The last two lenses are particularly important for graph-based disambiguation methods used in NEL, therefore they can become valuable debugging tools.

The naming of events and stories has been an issue, as often some narratives have an incidental name at the beginning, which due to lack of information is associated with the city in which the incident took place (e.g. Fire in London, West London Fire), whereas later it can morph into a more precise name that might include even the name of the street (e.g. Grenfell Tower Fire or simply Grenfell Tower). We generally kept the mainstream names in this corpus, but sometimes early information is included as

[15]http://scikit-learn.org/stable/modules/clustering.html
[16]Tweet Downloader by ID example: https://gist.github.com/giacbrd/b996cfe2f1d24752f23bd119fdd678f2

Table 11: Basic Statistics.

Type	News	Subs	Tweets
Documents	100	100	200
Entities	975	899	776

Table 12: Difference in number of entities between lenses.

Type	News	Subs	Tweets
Gold	975	899	776
Long	960	891	759
Embedded	1054	942	828

well, therefore the corpus can be a good start for anyone interested in how such stories develop over time and various media formats.

An overview of the initial lenses as reflected through the counts of entities can be found in Table 12. Early results of the Recognyze evaluation on this dataset can be found in Table 13. As it can easily be seen, the number of entities and the results for gold and long lenses are quite similar, whereas the number of entities and the evaluation results for the embedded lens tends to be higher. It has to be noted that while this is indeed the case, we generally tend to adhere to the convention of using the long lenses in practice. Embedded lenses add extra entities due to the fact that an entity like New York Giants Stadium would be counted as two entities in an embedded settings (New York and New York Giants Stadium). We will continue to add lenses and perform evaluations with them until we find the best settings for creating new annotations.

The geolocation dataset (StoryLens) created for this year's evaluation will be integrated into GERBIL and further developed to enable event-detection and several other types of tasks.

6.3.2 Evaluation

The following datasets were used extensively during the different evaluations we have performed during the last year:

- StoryLens (English). This dataset was described in the previous section. This dataset contained documents from multiple types of sources (e.g. tweets, subtitles, news articles) collected during the early months of the Summer 2017.
- LDL-2016 (English) - This Twitter corpus was annotated using Geonames and also contains POIs, place qualifiers (PQ) and other types of place descriptors whereas we were only interested in PNPs (Proper Name Places), as the rest of the entities had no links in Geonames currently anyway. This corpus contains locations from a specific area during a specific period of time (Hawaiian Islands during 2 hurricanes in 2014). The evaluation was done stricly to understand the performance of Recognyze in July 2016 for detecting GPEs, as previous evaluations only focused on PER and ORG. It contains multiple types of entities for location, but all entities are labeled with the same type: GPE, therefore a National Park or a building will not have different types. If you need to see the LOC, FAC, GPE types check the equivalent DBpedia, Wikidata or Geonames types.
- N3 Corpora (Reuters128 - English, News100 - German, RSS500 - English). The collection contains 3 datasets that were collected from different sources (newswire from the classic Reuters corpora, German news articles, RSS feeds). While these datasets were not focused only on locations, we have only used their geolocation entities in evaluations.

Table 13: Recognyze StoryLens evaluation with lenses.

Type	P	R	F1
Gold	0.36	0.38	0.37
Long	0.36	0.38	0.37
Embedded	0.41	0.40	0.41

Table 14: Location detection performance between two versions of Recognyze

Dataset	Version	P	R	F1
LDL-2016	v1.0	0.26	0.48	0.30
	v1.0b	0.33	0.50	0.39
N3-Reuters128	v1.0	0.61	0.59	0.60
	v1.0b	0.64	0.61	0.62

Table 15: Location detection performance - comparison with similar tools

Dataset	Language	Tool	P	R	F1
KORE50	EN	AIDA (ws)	0.25	0.31	0.28
		Babelnet	0.21	0.19	0.20
		Spotlight	0.30	0.31	0.30
		Recognyze	**0.35**	**0.37**	**0.36**
N3-Reuters128	EN	AIDA (ws)	0.39	**0.71**	0.50
		Babelnet	0.37	0.64	0.46
		Spotlight	0.42	0.66	0.51
		Recognyze	**0.64**	0.61	**0.62**

We have only selected the entities that belong to the three main location classes (LOC, GPE, FAC) and were linked to DBpedia entries. Since all the tools currently provide more than just NEL services and are rather general information extraction (IE) tools, this choice guarded each tool against bad results obtained due to a lot of lesser known entities picked up by their algorithms (e.g. a classic NEL tool might only provide results for named entities, whereas IE tools will provide any entities including ones that are not necessarily named and easy to identify, therefore the number of results is generally much higher in an IE setting). We have used the same settings like in the previous year of the project in order to be able to compare with the historical results.

NIL Clustering is an important part of NEL evaluations, and evaluation packages like GERBIL (Usbeck et al., 2015) or neleval (Hachey, Nothman, & Radford, 2014) do include it. The latest version of Recognyze provides integration with the Stanford tagger and this allows us to perform NER separately than NEL therefore enabling NIL clustering. However, since none of the tools we have compared provide such results we have chosen to include only the entities that pointed to DBpedia URIs. The choice of Stanford tagger for NER allowed us to double-check results, therefore even though its usage is not necessarily important when reporting evaluation results, it has served as a good tool for debugging some of the difficult cross-type disambiguation errors.

The same KB build (DBpedia 2015-10) was used in the evaluations in order to make it easier to compare with results from previous years. Overall performance was improved compared to Year 1, but focus was placed on obtaining better results on short texts as it can clearly be seen from the previous tables. The results from the Reuters corpus can only be improved up to a certain limit, due to various errors that exist in the KB (e.g. wrong mappings) or in the corpora itself (e.g. wrong annotation). We could use non-standard evaluation settings in order to bypass such errors, but we chose not to, and instead focus on evaluation settings that conform with the current standards set by the scientific community.

Most of the tools are optimized to provide better recall instead of better precision, but since Recognyze results are well-integrated with a set of front-end components, our focus has generally been on improving precision.

We have further developed several techniques for analyzing errors discovered during evaluations in order to continuously improve our module. Since some of the errors discovered were outside our control (e.g. evaluation tool errors, Knowledge Base errors), we have mostly focused on the errors that we could eliminate through further improvements (e.g. single or cross-type disambiguation errors, wrong annotations).

While today most of the evaluation work related to NEL is done using GERBIL (Usbeck et al., 2015), the main standard still remains the one proposed by NIST during the TAC KBP challenges (in the EDL tasks), therefore in order to collect and analyze our results we have used this format and the associated tools (e.g. neleval scorer)(Hachey et al., 2014). TAC-KBP results can be processed to obtain a primary error analysis limited to the validity of the results (e.g. results are marked as wrong link, extra link or missing). While such information is valuable, adding a semantic layer on top of it can offer researchers and developers the information they need in order to improve their tools. In order to capture the logical reasoning that has produced the error we have considered the following error types:

– **KB:** A Knowledge Base error is an error discovered in a particular KB version (e.g. DBpedia 2015-10);

– **DS:** Dataset errors contain pointers to the gold standard version used during an evaluation;

– **AN:** Annotator errors refer to the output of the classic NEL phases (e.g. NERC, linking, relation extraction, or graph disambiguation);

– **NIL:** NIL Clustering errors refer to the output of the NIL Clustering components;

– **SE:** Scorer or evaluation errors explain the errors reported by an evaluation script when the AN and DS outputs are similar.

A typical **KB** error[17] looks like the entity *de.dbr:2009* which has been marked as a location in the German DBpedia version 2015-10. The **DS** errors are generally cases of wrong annotations due to various causes: typos (we found many cases in which dots were missing from geographic abbreviations), a different language than the target one (e.g. German DBpedia instead of English), partial matches (e.g. geo entities missing parts of their name). **AN** error causes include abbreviation conflicts (e.g. for *Kent.* abbreviation, the annotator returns *Kent, UK* instead of *Kentucky*), same-type disambiguation errors, cross-type disambiguation errors (e.g. when an entity with a different type is returned), or generic terms (e.g. when words like *ship* or *Admiral* are returned instead of the real entities that are near them). **NIL** Clustering errors include entity mentions being shared among multiple clusters (e.g. role/title appearing in a different cluster than the name). While **SE** errors are not as frequent like the other categories, a classic example is represented by correct redirects. It has to be noted that in some cases an error can appear due to multiple causes (e.g. a partial match causes a different entity to be returned by the NEL system due to a wrong KB redirect - this case offering both an Annotator and a KB error).

By using this format which included the type and the cause of each error, we were able to continuously improve Recognyze during the last year. The format itself was created in September 2016, but was only introduced in our testing cycles gradually starting with March 2017.

6.4 API layer and integration with InVID

Recognyze is integrated into the InVID Platform, therefore it is used by default to annotate documents collected by the Data Ingestion Pipeline (InVID WP2) or uploaded to the platform via API (e.g. from the InVID Verification Application). The Recognyze API has a Swagger interface available for each server on which Recognyze and associated components (KB builds, web services,) are installed.

It is now also called via a private API to provide the InVID Context Aggregation and Analysis Service results with location data (see Section 7) so that geolocation-based contextualisation can be supported outside of the platform workflow, e.g. directly for a document in the Verification Application.

[17] KB errors mentioned in this text are from DBpedia 2015-10 release.

7 Context Aggregation and Analysis

During the second project year, we dedicated effort into extending the Context Aggregation and Analysis module to cover more online video sources, specifically Twitter and Facebook. This entailed analyzing the information provided by these sources and developing corresponding versions of the module outputs, which are similar to each other so as to allow a uniform UI appearance and structure, but are simultaneously customized for each type of video. Further improvements were the integration with the Location Detection module, an extension of our comment analysis to better detect verification-related comments in multiple languages, and a first attempt to develop a video verification algorithm that will utilize contextual information to automatically provide an estimate on the credibility of the video. This section presents these improvements, as well as the evaluation results of our automatic approach.

7.1 State of the art

The process which, within InVID, we have termed “contextual aggregation and analysis”, has been a common news verification practice ever since UGC started becoming an essential component of news reporting. In order to confirm or reject any news item (text, photo, video), reporters will typically resort to practices such as attempting to identify the first user that posted the item and ideally contacting them, or looking for inconsistencies between contextual characteristics of the posted item (date, location) and external knowledge about the event. The CAA component of InVID aims at organizing this part of the verification pipeline into an integrated tool.

The first attempts to replace ad hoc initiatives with a structured framework essentially consisted of guides and tutorials on what pieces of information are integral for verification, and how to exploit various existing online tools such as Google search, reverse image search, or Street View, for the purposes of verification. One of the most complete such handbooks, which is highly relevant still today, is The Verification Handbook by the European Journalism Centre, edited by Craig Silverman[18]. Google News Lab provides its own set of tutorials on how to use Google tools for verification[19]. The online investigative organization Bellingcat also provides its own guides, including one specific for UGC Videos [20].

Existing guidebooks are based on utilizing multiple different existing services. Attempts to bring together multiple functionalities into an integrated tool are currently limited. Amnesty International’s ”YouTube Data Viewer”[21] returns the video upload time/date, plus a number of thumbnails (extracted by YouTube) with links to Google reverse image search. Enrique Piracés’ Video Vault[22] allows archiving online videos to save them from being taken down, and provides information in three “component parts”: thumbnails, the video metadata as it appeared online, the video footage and the audio. It also provides a meeting room where multiple users can share these components and discuss them in real time. It also allows links for reverse image search on the thumbnails, and a toolbar to slow down playback, speed it up, zoom in on particular areas, rotate the video, and take a snapshot of particular moments. In a relevant field, TruthNest[23] provides an integrated solution for Tweet verification using contextual information.

However, these tools only manage to integrate a small fraction of the actual steps that an investigator may take when verifying a video. Analysis of video comments, location detection, profiling of the uploader’s account (e.g. how old it is, how many posts in the past), weather analysis, and analysis of the presence of the video in online media are only a few of the pieces of information that may be exploited for a comprehensive verification process. The aim of InVID Context Aggregation and Analysis module has been to move beyond existing limited or fragmented tools, and provide an integrated solution that covers multiple investigator requirements in an intuitive and user-friendly fashion. One feature that could prove helpful, and is currently not provided by any competing platform, is the algorithmic analysis of the available information, with the aim of providing an automatic assessment on the authenticity of the video. While –to our knowledge– there currently is no such attempt for videos, there are relevant tweet verification approaches in the literature (Gupta, Lamba, Kumaraguru, & Joshi, 2013; Boididou et al., 2015a, 2017). Such methods typically extract a number of linguistic and statistical features from the tweet text and the user account that posted it, and then train a classifier on a large dataset from past events to

[18]https://firstdraftnews.com/curriculum_resource/the-verification-handbook/
[19]https://newslab.withgoogle.com/course/verification
[20]https://www.bellingcat.com/resources/how-tos/2017/06/30/advanced-guide-verifying-video-content/
[21]https://citizenevidence.org/2014/07/01/youtube-dataviewer/
[22]https://www.bravenewtech.org/
[23]http://www.truthnest.com/

distinguish between fake and real tweets. The task is particularly relevant to our own needs, and so we decided to follow a similar methodology towards automatic video verification.

7.2 Method description

The InVID CAA module collects and analyzes the online context of videos with the aim of providing the analyst with pieces of information that may prove helpful for verification. Similar to other InVID modules, the CAA version developed during Year 1 of the project was able to analyze video UGC posted exclusively on the YouTube platform. However, in reality, news consumers get their information from a much broader variety of social media platforms, and correspondingly, news-related UGC videos may appear on any of them. According to a recent survey from the Pew Research Center[24], two-thirds (67%) of Americans report that they get at least some of their news on social media. Facebook by far leads as a source of news among social media with 45% and YouTube has the second highest percentage with 18%, followed by Twitter with 11%. Other social media sources with lower percentages include Instagram, Snapchat, LinkedIn, Reddit, WhatsApp and Tumblr.

Following these observations, during the second project year we dedicated effort towards extending the service to other video platforms and specifically Facebook and Twitter, since, together with YouTube they cover a significant majority of cases. The contextual cues that are collected or calculated to create the verification report follow the same structure as the one presented in D3.1. In short:

– Information obtained directly from the video source API.

– Information computed by the service using the raw data from the video source or elsewhere.

– Tweets sharing the target video.

– Weather conditions at the time and place where the video was supposedly captured.

With respect to YouTube videos and the information we extract directly from the YouTube API (e.g. video title, video description, video comments, etc.), we follow the same structure as the one presented in D3.1. However, several improvements were made during the second year, with respect to the clues we compute from this information. One change is that the Recognyze module presented in Section 6 was integrated with CAA, and has now replaced the Stanford Named Entity Recognizer (NER) we used during Year 1. The Recognyze algorithm provides much more accurate "mentioned locations" results. Another change in information analysis concerns the automatic detection of verification-related comments. In the new version, the list of verification words used to label video comments as *verification comments* was significantly extended, both in terms of the number of keywords and the number of supported languages. The module now identifies words in English, German, Greek, Arabic, French, Spanish, Farsi, and we intend to extend coverage to other languages as well (such as Russian), to cover the maximum number of video comments possible. Another change is that the thumbnail-based reverse image search, used for finding near duplicate images and videos, has been expanded to generate search URLs for the Yandex Image Search engine, besides Google Image Search.

The major change in the module, however, concerns the extension of coverage from YouTube into Facebook and Twitter. In contrast to other modules (e.g. Logo Detection, Near-Duplicate Detection) where such an extension was relatively straightforward, for CAA this required dedicated effort and planning. For other modules, coverage extension only required a way to locally acquire the video file from these additional platforms. For CAA, we had to evaluate whether there was correspondence between the information provided by Facebook/Twitter and the information provided by YouTube, and what adaptations would be required in the fields returned by the module.

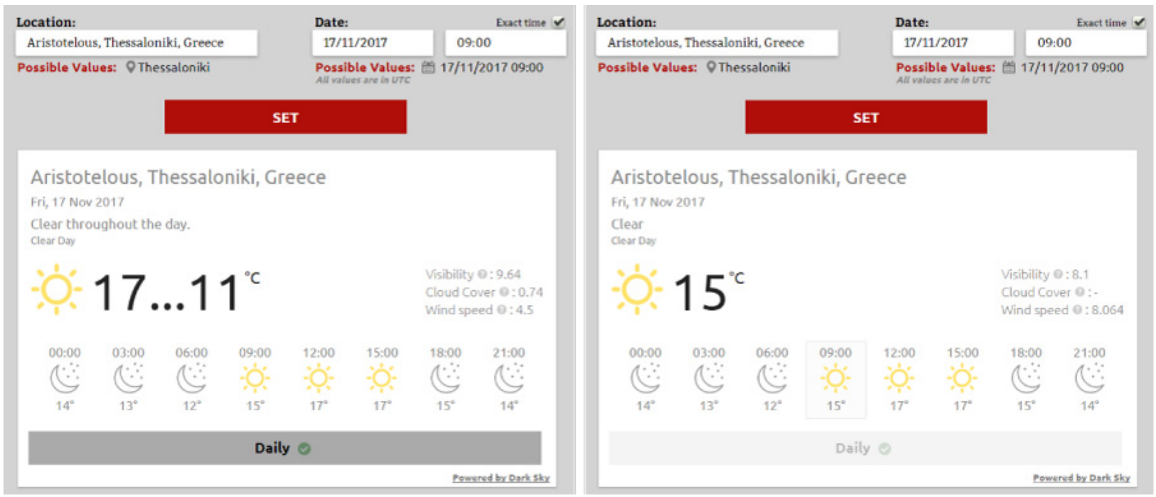
With respect to Facebook, the Graph API is the primary way to get data in and out of Facebook's social graph. There are essentially three types of accounts that may post a video: i) "Facebook User", representing a person on Facebook, ii) "Facebook Page" corresponding to a community, organization, business or other non-person entity, and iii) "Facebook Group", representing a page where multiple users may post. Facebook User is a restricted type and no information can be retrieved for videos posted by this type. For the other two types, a volume of information is provided by the API, and the module then proceeds to filter the pieces that are relevant for verification. The selected metadata for Facebook are presented in Table 16. It is worth noting that the set of elements that describes a Group where a video is posted is different than the one describing a Page. This difference arises from the difference in the purpose of each Facebook account type. The TYPE column of Table 16 indicates

[24]`http://www.journalism.org/2017/09/07/news-use-across-social-media-platforms-2017/`

whether the element refers to a group or page. The *video_description_mentioned_locations* field contains the locations mentioned of the video as they are returned by submitting the video description to the Recognyze module. Regarding tweets sharing the Facebook video, we have experimentally observed that it is in general not such a common case as with YouTube videos. Nonetheless, the module searches for tweets sharing the Facebook video and if they exist, it creates a Twitter timeline similar to the one created for YouTube videos. The tweets sharing the video are then submitted to the Tweet Verification Assistant API in order to be labeled as “fake” or “real”.

With respect to Twitter, videos embedded in a tweet, also known as *native Twitter videos*, are now supported in the latest release of Context Aggregation and Analysis service. A native Twitter video can provide information about the video itself and the user who shared it derived by the Twitter API, similar to YouTube and Facebook videos. The information is then filtered by the module. Table 17 presents the fields that the module returns for verification analysis. In contrast to Facebook and YouTube, Twitter does not contain user comments on posted videos. However, tweet replies can be considered to serve the same purpose. Consequently, tweet replies and the accompanying metadata about the time the reply was created, along with the user profile that shared the tweet are presented to the end user in the corresponding fields, and are analyzed in the same manner that comments are. This means that *verification replies* is created as the subset of the tweet replies that contain any of the listed verification words. The *tweet_text_mentioned_locations* and *user_description_mentioned_locations* fields contains locations extracted from the tweet text and user description text respectively. Additional information is provided by submitting the tweet to Tweet Verification Assistant API which responds with a verification label of fake or real indicating the credibility of the tweet itself. In order to build the Twitter timeline, which in the case of Facebook and YouTube videos is created from the tweets sharing the video, the video retweets are retrieved, and information about the users are presented into the timeline. The Tweet Verification Assistant API (Boididou et al., 2017) is then used to label these tweets as credible or not. In this way we have ensured that in the end the resulting output is quite similar for videos posted in any of the three platforms, despite their inherent differences.

Another change to the CAA module concerns the historical weather report feature. Weather reporting was introduced as a verification tool in D3.1, by integrating a third-party service. Following user comments during the second year, the report produced by the CAA module was extended to show, apart from the weather conditions of the time that the event supposedly happened, a detailed report per hour of the entire day. In Figure 16 (left), the minimum and maximum temperatures of the specified day are shown on the top of the picture together with a short summary (*Clear throughout the day*) and additional data such as the visibility, cloud cover and wind speed. The exact temperatures per hour are listed below, grouped by three hours, together with indicative images. If the exact time is known the corresponding weather conditions for that hour are shown (see Figure 16 (right). The module returns the data in JSON format and the visualization of Figure 16 is a screenshot of the CAA User interface `http://caa.iti.gr/`.


Location: Aristotelous, Thessaloniki, Greece
Possible Values: Thessaloniki
Date: 17/11/2017 09:00 Exact time ✓
Possible Values: 17/11/2017 09:00
All values are in UTC

Aristotelous, Thessaloniki, Greece
Fri, 17 Nov 2017
Clear throughout the day.
Clear Day
17...11 °C
Visibility : 9.64
Cloud Cover : 0.74
Wind speed : 4.5

00:00	03:00	06:00	09:00	12:00	15:00	18:00	21:00
14°	13°	12°	15°	17°	17°	15°	14°

Location: Aristotelous, Thessaloniki, Greece
Possible Values: Thessaloniki
Date: 17/11/2017 09:00 Exact time ✓
Possible Values: 17/11/2017 09:00
All values are in UTC

Aristotelous, Thessaloniki, Greece
Fri, 17 Nov 2017
Clear
Clear Day
15 °C
Visibility : 8.1
Cloud Cover : -
Wind speed : 8.064

00:00	03:00	06:00	09:00	12:00	15:00	18:00	21:00
14°	13°	12°	15°	17°	17°	15°	14°

Figure 16: The new weather report display of the CAA module.

The final methodological contribution we made during Year 2 was an experimental study towards a novel system for automatic video verification. In the work presented in D3.1, the CAA module essentially

Table 16: Fields returned by the Context Aggregation and Analysis module API for Facebook Videos

Indicator	Description	
Information about the video itself		
video_id	The ID of the video	
title	The video title or caption.	
content_category	The content category of the video.	
content_tags	Tags that describe the contents of the video.	
created_time	The time the video was initially published.	
updated_time	The last time the video or its caption was updated.	
embeddable	Whether the video is embeddable.	
video_likes	Video likes (People who liked the video)	
length	Duration of the video.	
picture	The URL for the thumbnail picture of the video.	
privacy	Privacy setting for the video.	
Comments		
video_comments	Text of comment	
video_author_comments	Name of User who commented	
video_author_url_comments	Link to User timeline who commented	
total_comment_count	Number of users comments for the video	
Page or Group that posted the video		**TYPE**
from	The profile that posted the video.	Page/Group
from_about	The "about" field of the profile that posted the video.	Page
from_category	The category of profile that posted the video.	Page
from_link	The Page's Facebook URL.	Page
from_fan_count	The fan count of profile that posted the video.	Page
from_is_verified	If the profile that posted the video is verified.	Page
from_description	The description of profile that posted the video.	Page/Group
from_website	The website of the profile that posted the video.	Page
from_location_city	The location (city) of profile that posted the video.	Page
from_location_country	The location (country) of profile that posted the video.	Page
from_cover	URL to the Photo that represents this cover photo.	Group
from_owner	The profile that created this group.	Group
from_updated_time	The last time the group was updated	Group
from_privacy	The privacy setting of the group. Possible values: CLOSED, OPEN, SECRET	Group
Information computed by the module		
verification_comments	Comments that contain verification words.	
num_verification_comments	Number of verification comments	
reverse_image_thumbnails_search_url_google	URLs of the reverse image thumbnails search for Google Image Search	
reverse_image_thumbnails_search_url_yandex	URLs of the reverse image thumbnails search for Yandex Image Search	
video_description_mentioned_locations	The locations mentioned in the video title and video description.	

took existing information from various sources, aggregated and restructured it, and presented it to the investigator for analysis. With the exception of the tweet verification feature, which indirectly provided an estimate on the credibility of the video by analyzing the credibility of the tweets reposting it, it did not offer any automated verification results on the video itself. During Year 2 of the project, we experimented towards developing a video verification system that could provide the investigator with a direct estimate of whether the video itself is likely "real" or "fake".

Our work towards an automatic verification approach was based on established tweet verification approaches from the literature, which analyze tweets using text and user features. Specifically, we based our approach on (Boididou et al., 2017), a tweet verification approach trained on a dataset called the Image Verification Corpus. The Image Verification Corpus consists of tweets from 53 past events, 7,229 of which have been labeled real and 10,628 fake, and was used for the MediaEval Verifying Multimedia Use tasks (Boididou et al., 2015b).

We attempted to devise a similar approach for YouTube videos. The information we decided to exploit was a) the comments accompanying the video, from where we would be able to extract linguistic features similar to the ones we can extract from tweets, and b) the video description and user account information, which is in many aspects similar to a Twitter user account information.

In order to extract comment features, we took the tweet features used by (Boididou et al., 2015a) and kept a subset of them relevant to the current task, as listed in Table 18, under "Comment-level features". Features such as #retweets and #hashtags were excluded since they are not applicable to YouTube

Table 17: Fields returned by the Context Aggregation and Analysis module API for Native Twitter Videos

Indicator	Description
	Information about the video itself
id_str	The string representation of the unique identifier for this Tweet.
full_text	The actual UTF-8 text of the status update.
created_at	UTC time when theTweet was created.
source	Utility used to post the Tweet as an HTML-formatted string. Tweets from the Twitter website have a source value of web.
retweet_count	Number of times this Tweet has been retweeted.
favorite_count	Indicates approximately how many times this Tweet has been liked by Twitter users.
hashtags	Represents hashtags which have been parsed out of the Tweet text.
urls	Represents URLs included in the text of a Tweet.
user_mentions	Represents other Twitter users mentioned in the text of the Tweet.
lang	When present, indicates a BCP 47 language identifier corresponding to the machine-detected language of the Tweet text or “und” if no language could be detected.
media_url	An http:// URL pointing directly to the video thumbnail file.
video_info_aspect_ratio	The aspect ratio of the video as a simplified fraction of width and height width:height
video_info_duration	The length of the video
video_info_url	The URL to the video file
	Comments
replies	Tweet replies
replies_name	The name of the user who posted the reply
replies_screen_name_url	URL to the profile of the user who posted the reply
replies_created_at	UTC time when this reply was created.
	User who posted the Tweet containing the video
name	The name of the user as theyve defined it. Not necessarily a persons name. Typically capped at 20 characters but subject to change.
user_screen_name	The screen name, handle, or alias that this user identifies themselves with. screen_names are unique but subject to change.
user_location	The user-defined location for this accounts profile. Not necessarily a location nor machine-parseable. This field will occasionally be fuzzily interpreted by the Search service
user_url	A URL provided by the user in association with their profile.
user_description	The user-defined UTF-8 string describing their account.
user_protected	When true, indicates that this user has chosen to protect their Tweets.
user_verified	When true, indicates that the user has a verified account.
user_followers_count	The number of followers this account currently has. Under certain conditions of duress, this field will temporarily indicate 0.
user_friends_count	The number of users,this account is following (AKA their followings). Under certain conditions of duress, this field will temporarily indicate 0.
user_listed_count	The number of public lists that this user is a member of.
user_favourites_count	The number of Tweets this user has liked in the accounts lifetime. British spelling used in the field name for historical reasons.
user_statuses_count	The number of Tweets (including retweets) issued by the user.
user_created_at	The UTC datetime that the user account was created on Twitter.
user_profile_image_url_https_original	A HTTPS-based URL pointing to the users profile image original size
user_profile_image_url_https_normal	A HTTPS-based URL pointing to the users profile image normal size
user_profile_image_url_https_mini	A HTTPS-based URL pointing to the users profile image mini size
user_profile_image_url_https_bigger	A HTTPS-based URL pointing to the users profile image bigger size
user_lang	The BCP 47 code for the users self-declared user interface language. May or may not have anything to do with the content of their Tweets.
	Information computed by the module
reverse_image_thumbnails_search_url_google	URLs of the reverse image thumbnails search for Google Image Search
reverse_image_thumbnails_search_url_yandex	URLs of the reverse image thumbnails search for Yandex Image Search
tweet_text_mentioned_locations	Locations mentioned in tweet text
user_description_mentioned_locations	Locations mentioned in user description text
Tweet_verification_label	Credibility label

video comments. Video verification was conducted on a two-level classification architecture. The Image Verification Corpus was used to train a first-level tweet/comment classifier. For each video, the trained model is then used to classify all video comments of a video, producing a score in the range of [0, 1] for each comment. Subsequently, we extract the 10-bin histogram of the comment verification values, which serves as a video-level descriptor for the second-level classifier (CL^{com}).

The second set of features, drawn from the video description and channel, was derived from the video metadata of the YouTube videos such as linguistic analysis of the video description, number of likes/dislikes etc. Thirteen contextual features are extracted based on the video description and four on the channel description, as shown in Table 18. A third set of features was then formed by concatenating the 10-bin histograms of the comment-level features and the video-level features.

7.3 Evaluation and progress since Year 1

During the second year, our evaluations concerned our approach for automatically analyzing the features extracted by the module to assess whether a video is real or fake. We also performed an analysis on the presence of verification-related comments in videos given the improved detection list we created. Finally, we improved the service performance following the feedback from the test and validation cycles.

Concerning automatic video verification, an evaluation was conducted on all three proposed models (comment-based, video/channel-based, concatenation) using the Fake Video Corpus (FVC) described in Section 2.2. The dataset consists of 227 UGG videos (117 fake and 110 real ones) taken from the YouTube platform, and covers all known types of fake video content, namely staged videos, videos in which the context of the depicted events is misrepresented, past videos claiming to present recent news, video or audio content altered by editing, and computer-generated Imagery (CGI) posing as real. The FVC dataset was split into training and test sets using 10-fold crossvalidation to evaluate the performance of the approaches. A Radial Basis Function Support Vector Machine (RBF SVM) classifier was then trained and evaluated on the FVC dataset. In Table 19, we present the results for the two individual features (first two rows of the table), and the increase in terms of all evaluation measures for the concatenated features (CL^{cat}).

We further tested an agreement-based method (Boididou et al., 2017) in which we compare the outputs of CL^{com} and CL^{vid} and divide the evaluated videos into two subsets, the *agreed* subset which contains the videos that both CL^{com} and CL^{vid} labeled as fake or real, and the *disagreed* subset where each model has assigned a different label. We kept the labels of the *agreed* videos, and investigated two techniques to re-classify the *disagreed* subset. In the first technique, the model build using the concatenated features CL^{cat} was used to re-classify the disagreed dataset, resulting in an improvement in accuracy (Agreement CL^{cat} row of Table 19. The second technique assumes that the agreed labels are correct with high likelihood and aggregates them to the initial training set, in order to retrain the concatenated model CL^{cat}. The agreement-based retraining approach is presented in Figure 17 Both methods reached F1-measure close to 0.80, that is 8% higher that the initial models.

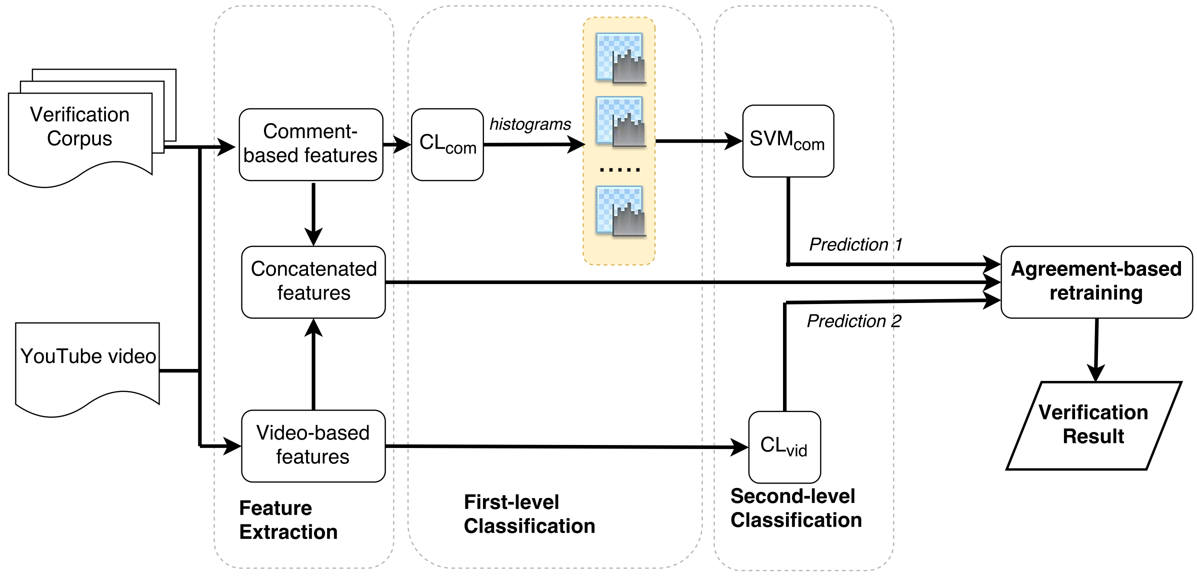


Figure 17: Overview of the video verification agreement-based retraining approach.

Finally, we examined a case of ideal fusion of the aforementioned classifiers. In ideal fusion, we assume to have a perfect method that always selects the classifier which is correct, thus a wrong classification occurs only when both classifiers are wrong. The third row of Table 19 shows the significant increase in performance when combining the results of both classifiers and highlights the need to utilize a more robust fusion method in order to provide investigators with an integrated estimate on the veracity of the video.

Overall, these experiments demonstrate the potential of machine learning approaches for automatic video verification. However, the current evaluations were rather limited and can only be considered indicative. In the near future, we intend to continue with more evaluations (to the extent that we can collect relevant annotated video content). Furthermore, it is currently unclear what is the best way to

Table 18: Comment and Video - level features

Comment-level features	**Video-level features**	
	From video description	**From channel description**
Text length	Text length	Channel view count
#words	#words	Channel comment count
Contains question mark	Contains question mark	Channel subscriber count
Contains exclamation mark	Contains exclamation mark	Channel video count
Contains happy emoticon	Contains 1st person pronoun	
Contains sad emoticon	Contains 3rd person pronoun	
Contains 1st person pronoun	Number of uppercase characters	
Contains 2nd person pronoun	Number of positive sentiment words	
Contains 3rd person pronoun	Number of negative sentiment words	
Number of uppercase characters	Number of slang words	
Number of positive sentiment words	Has : symbol	
Number of negative sentiment words	#question marks	
Number of slang words	#exclamation marks	
Has : symbol		
Has ”please”		
#question marks		
#exclamation marks		
Readability score		

present the analysis results to the user, since it is not easy to explain the exact workings of the algorithm to a user, and justify why the algorithm reached a certain conclusion. The most productive way to present this feature within the InVID platform is being explored in cooperation with our user partners.

Table 19: Video Classification Results

	Precision	**Recall**	**F1**
Comments	0.77	0.67	0.72
Video metadata	0.80	0.68	0.73
Concatenation	0.87	0.70	0.78
Agreement CL^{cat}	0.70	0.87	0.79
Retrain CL^{cat}	0.72	0.86	0.79
Ideal fusion	0.90	0.95	0.92

Another set of changes since Year 1 concerned the InVID evaluations. During the InVID test and validation cycles, several bugs were detected, and furthermore -since CAA is a module whose usefulness is highly dependent on the user experience, many important suggestions for improvements were submitted by the users. The most important ones are listed below:

– The extension to other video sources than YouTube was a strong suggestion and a priority for us. Thus, at the second year release of the module we covered the most popular and used video platforms - YouTube, Facebook, Twitter.

– The testers noticed several alternatives to the video URLs. In that case, we created a list of all possible alternative URLs for each video source and we then implemented a preprocessing step that takes the video URL and recognizes the video source and video id which are the elements that are needed to start the process. Example YouTube URL formats:

  ◦ `http://youtu.be/MsrcqbQeLUA`
  ◦ `http://www.youtube.com/embed/http://youtu.be/MsrcqbQeLUA`
  ◦ `https://www.youtube.com/watch?time_continue=2&v=MsrcqbQeLUA`
  ◦ `https://www.youtube.com/watch?v=Q73CMzaHMAE&feature=youtu.be&t=40s`

  Example Facebook URL formats:

  ◦ `https://www.facebook.com/Dawahofislamuk/videos/796804540380035/`
  ◦ `https://www.facebook.com/groups/1822790977746257/permalink/1838155972876424/`

- There were fields present in the first versions of the CAA UI, of which the meaning was often not clear or misunderstood by many investigators. We included tooltips that explain the meaning and purpose of these fields in verification.
- Although the English language dominates in the UGC videos, there are a lot of other languages that are used in the video comments. We decided to extend the verification list not only to other languages but also with more verification-related keywords.
- Another useful point, concerning the service API, was the update of the processing status. Since certain processing steps may take a relatively long time, representative messages are now returned at each step of the process and, in the case of a request failure, error messages are given explaining the reason. These messages are shown in Tables 21, 22.

7.4 API layer and integration with InVID

A major update of the Context Aggregation and Analysis service is the extension of compatible video source platforms to Facebook and Twitter. The first release of the API required the video ID as input. However, the need of distinguishing the video source necessitated either to add a new parameter, which would provide the video source, or the change of the id parameter to the entire video URL. The latter solution is implemented in the current release of the module. The end user needs to provide the URL of a YouTube, Facebook or Twitter video at the main verification call `/verify_video` (first row of Table 20) to start the contextual verification assistance processes and a representative message is returned. We extended the module to return either status or error messages for all possible cases. A list with the calls exposed by the CAA module is presented in Table 20. The status messages are listed in Table 21 and the error messages are presented in Table 22. Corresponding messages exist for the Facebook and Twitter videos e.g. `FACEBOOK_COMPLETED_TWITTER_SHARES_PROCESSING` and `TWITTER_COMPLETED_TWITTER_SHARES_PROCESSING`.

Another major update of the module is the integration of the `reprocess` parameter. It is an optional parameter with default value equal to 0. The first time a video is submitted to the module all processes are executed and the retrieved information along with the calculated data are temporally stored on a local server. If the video is submitted again the responses are directly returned. However, there are elements accompanying the video that may have changed since the last call – this is a very common case when a video is submitted to the module right after it is publicly posted, and revisited a while later. The video comments/replies, the number of likes/dislikes, the number of tweets sharing the video are some of the elements that might have changed. The module by default searches for new information only if 24 hours have passed since the last call of a video. By adding the `reprocess` parameter at the URL call with value equal to 1 the module will be forced to check for new information.

Table 20: Calls exposed by the Context Aggregation and Analysis module API

Service	URL	Parameters		
		Required	Optional	Default values
Verify video	`/verify_video`	url=<video_URL>	reprocess=<1 or 0>	0
Video Verification Report	`/get_verification`	url=<video_URL>	-	-
Tweet Shares Report	`/get_twverification`	url=<video_URL>	reduced=<1 or 0>	0
Weather Report	`/weather`	time=<timestamp> location=<place>	-	-

Two separate calls are needed for accessing the collected and calculated data. The Video verification Report (Table 20) refers to all data about the video and the Tweet Shares Report (Table 20) returns information about the tweets sharing the video. An optional parameter named `reduced` is included in the Tweet Shares Report in order to return only the fields of the Tweet Objects that are of importance for creating the Twitter timeline (Figure 18).

Another major improvement integrated to the CAA API is the asynchronous execution of all processes. A field named `processing_status` is added at both responses (Video verification report and Tweet shares report) and informs whether the process is still in progress of has been completed. The end user can retrieve the data that have been collected or generated even if the entire processes have not finished. When the `processing_status` returns `done` the final, complete responses can be retrieved.

All features are visualized in a demo User Interface hosted at `http://caa.iti.gr/`. The user can add a URL of a video at ”Input Video” field and then click *Verify* to start the process (Figure 19).

Table 21: Status messages of main verification call of CAA API (/verify_video
)

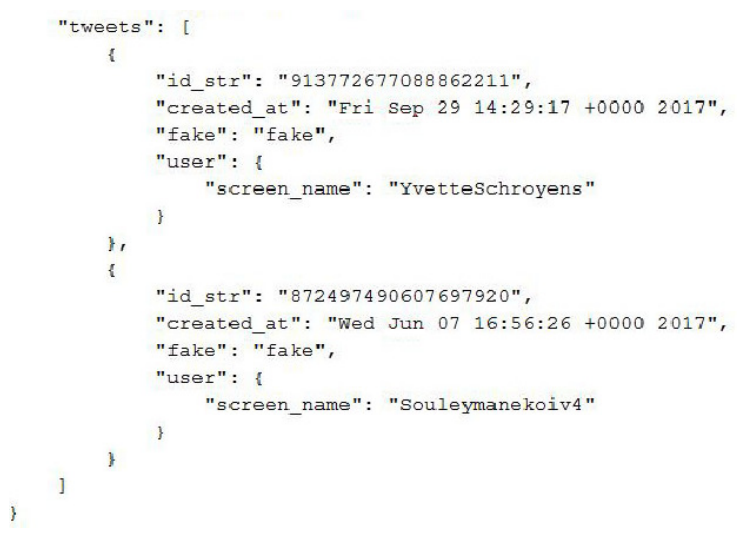
YOUR_REQUEST_HAS_BEEN_QUEUED: A video is submitted for first time.
VIDEO_PROCESSED_PREVIOUSLY_CHECK_FOR_NEW_INFORMATION_THRESHOLD_EXCEEDED: Video has already been submitted. If a threshold of 24 hours has been exceeded since the last call, the module checks if there is new information.
VIDEO_PROCESSED_PREVIOUSLY_CHECK_FOR_NEW_INFORMATION_REPROCESS_REQUESTED: Video has already been submitted. If the threshold of 24 hours has not been exceeded but the user want to force the module to check if there is new information.
PROCESSING: All functionalities are in progress
YOUTUBE_COMPLETED_TWITTER_SHARES_PROCESSING: YouTube processing has finished but Twitter shares processing is still in progress.
TWITTER_SHARES_COMPLETED_YOUTUBE_PROCESSING: Twitter shares processing has finished but YouTube processing is still in progress.
VIDEO_PROCESSING_DONE: All processes are finished. Overall verification results can be returned.

Table 22: Error messages of main verification call of CAA API (/verify_video
)

THIS_VIDEO_CANNOT_BE_FOUND: The video does not exist.
VIDEO_PROCESSED_PREVIOUSLY_BUT_REMOVED_FROM_YOUTUBE_ALL_METADATA_ARE_DELETED: The video has been submitted and processed previously but removed by the YouTube video platform. All metadata are removed.

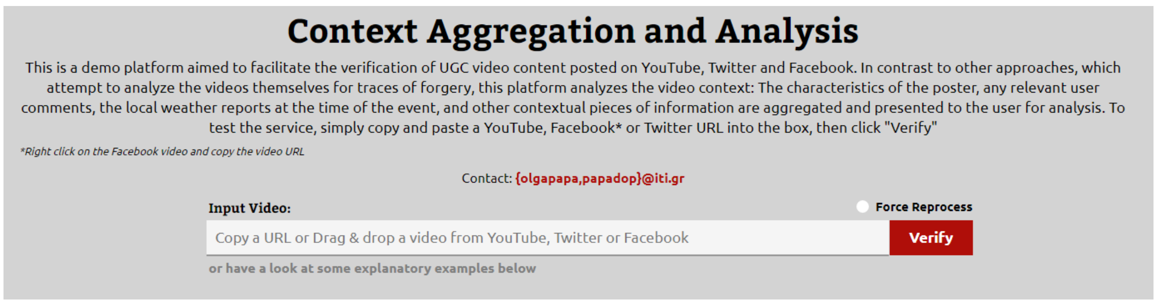
The verification data visualization is organized in sections. The *General* section contains the video and user elements, the *Comments* section presents the overall video comments and the verification ones separately, the *Thumbnail* section shows the retrieved thumbnails of the video accompanied with direct links to Google and Yandex image search. Following is the *Weather Context* section which is independent and can be triggered by providing a location and time -the fields are pre-filled by suggested values taken from the video metadata and the location detection results. At the bottom, the *Twitter Context* section presents the tweets sharing the video in a timeline format, and green or red color boxes highlight whether the tweet is estimated to be credible or not, based on our tweet verification algorithm. The listed sections appear directly after a video is submitted, and are either empty or containing part of the elements which have already been collected. The information is updated gradually until the final, complete output is returned.

With respect to the current stage of integration, the CAA API is integrated into the Verification App, and the InVID Verification Plugin. The Verification App collects all output produced by the CAA Service API. For the InVID Verification Plugin, a subset of the verification report created by the API is included at the 'analysis' tab of the Verification Plugin. In this case, mainly information derived directly by the video platform is reproduced. Regarding the output generated by the module features, the "mentioned locations" extracted by the video title and description are presented as part of the verification assistance. The thumbnails of the video are connected to a button, one for Google Image Search and another for Yandex Image Search, which submits the images for reverse image search. Correspondingly, a Twitter video search button triggers Twitter search and submits the video URL as query. The list of tweets sharing the video are retrieved but no further information – such as the verification labels extracted by the Tweet Verification Assistant API – is shown. Finally, the text of the video comments is omitted and just the number of comments and verification comments is listed at the 'analysis' tab of the Verification Plugin.



```
    "tweets": [
        {
            "id_str": "913772677088862211",
            "created_at": "Fri Sep 29 14:29:17 +0000 2017",
            "fake": "fake",
            "user": {
                "screen_name": "YvetteSchroyens"
            }
        },
        {
            "id_str": "872497490607697920",
            "created_at": "Wed Jun 07 16:56:26 +0000 2017",
            "fake": "fake",
            "user": {
                "screen_name": "Souleymanekoiv4"
            }
        }
    ]
}
```

Figure 18: Reduced tweet shares response.



Context Aggregation and Analysis

This is a demo platform aimed to facilitate the verification of UGC video content posted on YouTube, Twitter and Facebook. In contrast to other approaches, which attempt to analyze the videos themselves for traces of forgery, this platform analyzes the video context: The characteristics of the poster, any relevant user comments, the local weather reports at the time of the event, and other contextual pieces of information are aggregated and presented to the user for analysis. To test the service, simply copy and paste a YouTube, Facebook* or Twitter URL into the box, then click "Verify"

Right click on the Facebook video and copy the video URL

Contact: {olgapapa,papadop}@iti.gr

Input Video: ○ **Force Reprocess**

Copy a URL or Drag & drop a video from YouTube, Twitter or Facebook **Verify**

or have a look at some explanatory examples below

Figure 19: CAA User Interface URL input field

8 Outlook and next steps

In D3.2, we presented our progress in WP3 Tasks during the second year of the project. We presented our extension of the Fake Video Corpus, which reflects our analysis of real-world occurrences of fake and real UGC videos, and the perceived differences between them. We then presented the individual WP3 modules, the progress since Year 1 and their current performance evaluations and integration status. In all cases, the presented modules have integrated and improved state-of-the-art technologies, and have led to increased performance to the point of being ready for real-world, large-scale testing. The presented modules, covering the problem of video verification from multiple complementary aspects, are integrated with the InVID platform and in many cases also provide open, stand-alone, online demos for evaluation and dissemination. The end of the second year finds us with an integrated toolset more complete than any competing product in the market, and consisting of technologies that either match or surpass any competition in the academic state of the art with respect to real world applicability. At this stage, the verification aspect of the InVID platform offers an integrated solution that already would be a valuable tool for any investigator. We intend to dedicate the remaining third year to refining the performance of the modules to remain ahead of the state of the art while simultaneously adapting them to the needs of the end users.

Video forensics. *Our work in video forensics will remain confidential and has been removed from this version of the document.*

Near-duplicate detection. With respect to Near-Duplicate Video Retrieval, we developed a new approach which leverages the effectiveness of features extracted from intermediate convolution layers and Deep Metric Learning. We build a DML architecture based on video triplets and a novel triplet generation scheme that generates a compact video-level representation for the NDVR problem. The proposed approach was thoroughly tested on two CNN architectures and exhibits highly competitive performance when developed on an independent dataset from the evaluation set. Furthermore, it outperformed all compared approaches from the literature by a clear margin. Finally, the performance of the novel approach was compared with the method presented in D3.1 in terms of Precision-Recall and mAP and in two different setups of CC_WEB_VIDEO dataset and demonstrated significant improvement. In terms of the NDD service, we modified the DML method to facilitate our needs and set up a pipeline to index videos discovered from the trending topic detection component of WP2. Additionally, the API was updated to support any video source from the popular video platforms and fixed all bugs reported in the test and validation cycles. An annotation tool was developed for the creation of an evaluation dataset.

For the next year, our primary focus is the creation of the dataset in order to conduct more comprehensive evaluations of the developed approaches. We also plan to look into further improvements to the proposed approach, e.g. by considering more effective fusions schemes (compared to early and late fusion) and by training the DML architecture end-to-end (instead of using features from pre-trained CNN architectures). Moreover, we will also devise a better approach for Near-Duplicate Localization and we will assess its applicability on the problem of Partial Duplicate Video Retrieval (PDVR). Finally, it has been observed that there are a lot of cases where short scenes of movie trailers or making-of videos are presented as breaking-news. Hence, to be able to detect such cases, we plan to collect trailers and making-of videos of action movies and series in order to enrich our video database.

Logo detection. With respect to Logo Detection, we replaced the initial keypoint matching algorithm with a Region-proposal Convolutional Neural Network designed for object detection. We overcame the limitation in training data by devising a method to generate artificial training images. We demonstrated the method's superiority in terms of speed and scalability, and its comparable accuracy. Also, in contrast to the keypoint-based method, the new R-CNN approach has greater potential for improvement.

In the next year we will experiment with data augmentation techniques and training parameters to further improve performance. One idea is to introduce perspective transformations to the training data, to emulate logos that are not overlaid on the video but are located on depicted objects, e.g. clothes. This extension of the training approach will not only allow us to capture more logos, but will most likely increase the retrieval performance by allowing the network to capture the essence of the logo regardless of its shape and perspective. In terms of integration, we have now provided a new timeline-based output format which will make results more readable, and have also included the option for users to submit additional logos. With the help of the user partners, we will be collecting any submitted logos, aiming to provide a comprehensive logo database by the end of the project, alongside the final version of the

Logo Detection module.

Location detection. With respect to location detection, for Year 3 we plan to continue to improve the current geolocation functionality, but, in addition to this task, we will also focus on some of the other components from the Recognyze ecosystem that might help with this endeavour. One component integrates Recognyze with the classic Stanford NER solution, therefore allowing us to easily collect the NIL entities. Another component helps us visualize and evaluate the tool.

We plan to continuously develop, update to latest KB versions and evaluate several different methods of improving geolocation detection in the final deliverable, for example: i) basic functionality (e.g. SPARQL DBpedia queries); ii) SPARQL federation (e.g. DBpedia, Wikidata and Geonames queries together); iii) map-based geolocation; iv) geolocation and fine-grained addresses; v) all combined. The main purpose of doing this is to create a hybrid solution that will enable us to deliver one of the best semantic geolocation solutions available.

Context Aggregation and Analysis. The Context Aggregation and Analysis module was extended to analyze content from two additional sources, Facebook and Twitter. A number of adaptations to the displayed fields were made, to suit the format of the metadata produced by these two platforms. Furthermore, various adaptations to the provided information were made in response to the users' needs. We also presented our experiments towards designing an algorithm for automatic post verification using contextual features. We also introduced various changes to the service implementation and API, making it more fast and user-friendly.

In the final project year, besides dedicating effort to further improving the responsiveness of the service, we will further experiment with automatic verification approaches, with the aim of integrating into the contextual verification report, a value indicating our estimate of whether the video is fake or not. This will entail more extended experiments with video metadata descriptors, as well as the fusion of the results between the two classifiers presented. With respect to verification-related comments, we will further investigate the verification list and we intend to implement a more dynamic system of including verification words into the module.

Overall, at the end of the second year we find ourselves having developed and integrated a set of tools beyond the state of the art covering complementary aspects of the video verification process, and having demonstrated their superior performance and suitability for the tasks at hand. The next year will be spent refining and improving these tools, building upon our existing progress to deliver the final, integrated verification framework of InVID. Given that the current, updated verification framework already puts the InVID platform ahead of any current competition, further improvements and refinements are expected to lead to a unique innovative toolset that can assist investigators improve on current video verification procedures, and allow timely and accurate assessment of news-related video UGC in the form of a successful product.

References

Boididou, C., Andreadou, K., Papadopoulos, S., Dang-Nguyen, D.-T., Boato, G., Riegler, M., & Kompatsiaris, Y. (2015a). Verifying multimedia use at mediaeval 2015. In *Mediaeval* (Vol. 1436). CEUR-WS.org.

Boididou, C., Andreadou, K., Papadopoulos, S., Dang-Nguyen, D.-T., Boato, G., Riegler, M., & Kompatsiaris, Y. (2015b). Verifying multimedia use at mediaeval 2015. In *Mediaeval.*

Boididou, C., Papadopoulos, S., Zampoglou, M., Apostolidis, L., Papadopoulou, O., & Kompatsiaris, Y. (2017). Detection and visualization of misleading content on twitter. *International Journal of Multimedia Information Retrieval*, 1–16.

Cai, Y., Yang, L., Ping, W., Wang, F., Mei, T., Hua, X.-S., & Li, S. (2011). Million-scale near-duplicate video retrieval system. In K. S. Candan, S. Panchanathan, B. Prabhakaran, H. Sundaram, W. chi Feng, & N. Sebe (Eds.), *Proceedings of the 19th international conference on multimedia 2011, scottsdale, AZ, USA, november 28 - december 1, 2011* (pp. 837–838). ACM. Retrieved from `http://doi.acm.org/10.1145/2072298.2072484`

Chechik, G., Sharma, V., Shalit, U., & Bengio, S. (2010). Large scale online learning of image similarity through ranking. *Journal of Machine Learning Research*, *11*(Mar), 1109–1135.

Chen, D., Yuan, Z., Hua, G., Zheng, N., & Wang, J. (2015). Similarity learning on an explicit polynomial kernel feature map for person re-identification. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 1565–1573).

Chou, C.-L., Chen, H.-T., & Lee, S.-Y. (2015). Pattern-based near-duplicate video retrieval and localization on web-scale videos. *IEEE Trans. Multimedia*, *17*(3), 382–395. Retrieved from `http://dx.doi.org/10.1109/TMM.2015.2391674`

Chum, O., & Zisserman, A. (2007). An exemplar model for learning object classes. In *Computer vision and pattern recognition, 2007. cvpr'07. ieee conference on* (pp. 1–8).

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009a). Imagenet: A large-scale hierarchical image database. In *Computer vision and pattern recognition* (pp. 248–255).

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009b). ImageNet: A Large-Scale Hierarchical Image Database. In *Cvpr09.*

Douze, M., Jegou, H., & Schmid, C. (2010). An image-based approach to video copy detection with spatio-temporal post-filtering. *IEEE Trans. Multimedia*, *12*(4), 257–266. Retrieved from `http://dx.doi.org/10.1109/TMM.2010.2046265`

Ferrari, V., Fevrier, L., Jurie, F., & Schmid, C. (2008). Groups of adjacent contour segments for object detection. *IEEE transactions on pattern analysis and machine intelligence*, *30*(1), 36–51.

Girshick, R. (2015). Fast r-cnn. In *Proceedings of the ieee international conference on computer vision* (pp. 1440–1448).

Gu, C., Lim, J. J., Arbeláez, P., & Malik, J. (2009). Recognition using regions. In *Computer vision and pattern recognition, 2009. cvpr 2009. ieee conference on* (pp. 1030–1037).

Gupta, A., Lamba, H., Kumaraguru, P., & Joshi, A. (2013). Faking sandy: characterizing and identifying fake images on twitter during hurricane sandy. In *Proceedings of the 22nd international conference on world wide web* (pp. 729–736).

Hachey, B., Nothman, J., & Radford, W. (2014). Cheap and easy entity evaluation. In *Proceedings of the 52nd annual meeting of the association for computational linguistics, ACL 2014, june 22-27, 2014, baltimore, md, usa, volume 2: Short papers* (pp. 464–469). The Association for Computer Linguistics. Retrieved from `http://aclweb.org/anthology/P/P14/P14-2076.pdf`

Hadsell, R., Chopra, S., & LeCun, Y. (2006). Dimensionality reduction by learning an invariant mapping. In *Computer vision and pattern recognition, 2006 ieee computer society conference on* (Vol. 2, pp. 1735–1742).

Hao, Y., Mu, T., Hong, R., Wang, M., An, N., & Goulermas, J. Y. (2017). Stochastic multiview hashing for large-scale near-duplicate video retrieval. *IEEE Transactions on Multimedia*, *19*(1), 1–14.

Hellmann, S., Lehmann, J., Auer, S., & Brümmer, M. (2013). Integrating NLP using Linked Data. In H. Alani et al. (Eds.), *The semantic web - ISWC 2013 - 12th international semantic web conference, sydney, nsw, australia, october 21-25, 2013, proceedings, part II* (Vol. 8219, pp. 98–113). Springer.

Hoffart, J., Yosef, M. A., Bordino, I., Fürstenau, H., Pinkal, M., Spaniol, M., ... Weikum, G. (2011). 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* (pp. 782–792). ACL.

Huang, J., Kumar, S. R., Mitra, M., Zhu, W.-J., & Zabih, R. (1999). Spatial color indexing and applications. *International Journal of Computer Vision*, *35*(3), 245–268.

Huang, Z., Shen, H. T., Shao, J., Zhou, X., & Cui, B. (2009). Bounded coordinate system indexing for real-time video clip search. *ACM Trans. Inf. Syst*, *27*(3). Retrieved from http://doi.acm.org/10.1145/1508850.1508855

Ji, H., Pan, X., Zhang, B., Nothman, J., Mayfield, J., McNamee, P., & Costello, C. (2016). Overview of TAC-KBP2015 Tri-lingual Entity Discovery and Linking. In *Eighth text analysis conference (tac).* NIST.

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R. B., ... Darrell, T. (2014). Caffe: Convolutional architecture for fast feature embedding. In K. A. Hua, Y. Rui, R. Steinmetz, A. Hanjalic, A. Natsev, & W. Z. 0001 (Eds.), *Proceedings of the ACM international conference on multimedia, MM '14, orlando, FL, USA, november 03 - 07, 2014* (pp. 675–678). ACM. Retrieved from http://dl.acm.org/citation.cfm?id=2647868

Jiang, Y.-G., Jiang, Y., & Wang, J. (2014). Vcdb: a large-scale database for partial copy detection in videos. In *European conference on computer vision* (pp. 357–371).

Ke, Y., & Sukthankar, R. (n.d.). Pca-sift: A more distinctive representation for local image descriptors. In *Computer vision and pattern recognition, 2004. cvpr 2004. proceedings of the 2004 ieee computer society conference on* (Vol. 2, pp. II–II).

Kingma, D., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kordopatis-Zilos, G., Papadopoulos, S., Patras, I., & Kompatsiaris, Y. (2017a). Near-duplicate video retrieval by aggregating intermediate cnn layers. In *International conference on multimedia modeling* (pp. 251–263).

Kordopatis-Zilos, G., Papadopoulos, S., Patras, I., & Kompatsiaris, Y. (2017b, Oct). Near-duplicate video retrieval with deep metric learning. In *Web-scale vision and social media (iccv-w).*

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In P. L. Bartlett, F. C. N. Pereira, C. J. C. Burges, L. Bottou, & K. Q. Weinberger (Eds.), *Advances in neural information processing systems 25: 26th annual conference on neural information processing systems 2012. proceedings of a meeting held december 3-6, 2012, lake tahoe, nevada, united states* (pp. 1106–1114). Retrieved from http://papers.nips.cc/book/advances-in-neural-information-processing-systems-25-2012

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., & Dyer, C. (2016). Neural architectures for named entity recognition. In K. Knight, A. Nenkova, & O. Rambow (Eds.), *NAACL HLT 2016, the 2016 conference of the north american chapter of the association for computational linguistics: Human language technologies, san diego california, usa, june 12-17, 2016* (pp. 260–270). The Association for Computational Linguistics. Retrieved from http://aclweb.org/anthology/N/N16/N16-1030.pdf

Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., ... Bizer, C. (2015). Dbpedia - A large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, *6*(2), 167–195. Retrieved from https://doi.org/10.3233/SW-140134 doi: 10.3233/SW-140134

Liu, J., Huang, Z., Cai, H., Shen, H. T., Ngo, C.-W., & Wang, W. (2013). Near-duplicate video retrieval: Current research and future trends. *ACM Comput. Surv*, *45*(4), 44. Retrieved from http://doi.acm.org/10.1145/2501654.2501658

Liu, L., Lai, W., Hua, X.-S., & Yang, S.-Q. (2007). Video histogram: A novel video signature for efficient web video duplicate detection. In *International conference on multimedia modeling* (pp. 94–103).

Middleton, S. E., & Krivcovs, V. (2016). Geoparsing and geosemantics for social media: Spatiotemporal grounding of content propagating rumors to support trust and veracity analysis during breaking news. *ACM Trans. Inf. Syst.*, *34*(3), 16:1–16:26. Retrieved from http://doi.acm.org/10.1145/2842604 doi: 10.1145/2842604

Mignon, A., & Jurie, F. (2012). Pcca: A new approach for distance learning from sparse pairwise constraints. In *Computer vision and pattern recognition (cvpr), 2012 ieee conference on* (pp. 2666–2672).

Ozdikis, O., Oguztüzün, H., & Karagoz, P. (2017). A survey on location estimation techniques for events detected in twitter. *Knowl. Inf. Syst.*, *52*(2), 291–339. Retrieved from https://doi.org/10.1007/s10115-016-1007-z doi: 10.1007/s10115-016-1007-z

Paisitkriangkrai, S., Shen, C., & van den Hengel, A. (2015). Learning to rank in person re-identification with metric ensembles. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 1846–1855).

Plu, J., Rizzo, G., & Troncy, R. (2016). Enhancing entity linking by combining NER models. In H. Sack, S. Dietze, A. Tordai, & C. Lange (Eds.), *Semantic web challenges - third semwebeval challenge at ESWC 2016, heraklion, crete, greece, may 29 - june 2, 2016, revised selected papers* (Vol. 641, pp. 17–32). Springer. Retrieved from `https://doi.org/10.1007/978-3-319-46565-4_2` doi: 10.1007/978-3-319-46565-4_2

Radenović, F., Tolias, G., & Chum, O. (2016). Cnn image retrieval learns from bow: Unsupervised fine-tuning with hard examples. In *European conference on computer vision* (pp. 3–20).

Razavian, A. S., Sullivan, J., Carlsson, S., & Maki, A. (2016). Visual instance retrieval with deep convolutional networks. *ITE Transactions on Media Technology and Applications*, *4*(3), 251–258.

Ren, S., He, K., Girshick, R., & Sun, J. (2015a). Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems* (pp. 91–99).

Ren, S., He, K., Girshick, R. B., & Sun, J. (2015b). Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, *abs/1506.01497*. Retrieved from `http://arxiv.org/abs/1506.01497`

Rizzo, G., Pereira, B., Varga, A., van Erp, M., & Basave, A. E. C. (2017). Lessons learnt from the named entity recognition and linking (NEEL) challenge series. *Semantic Web*, *8*(5), 667–700. Retrieved from `https://doi.org/10.3233/SW-170276` doi: 10.3233/SW-170276

Samet, H., Sankaranarayanan, J., Lieberman, M. D., Adelfio, M. D., Fruin, B. C., Lotkowski, J. M., ... Teitler, B. E. (2014). Reading news with maps by exploiting spatial synonyms. *Commun. ACM*, *57*(10), 64–77. Retrieved from `http://doi.acm.org/10.1145/2629572` doi: 10.1145/2629572

Schroff, F., Kalenichenko, D., & Philbin, J. (2015). Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 815–823).

Shang, L., Yang, L., Wang, F., Chan, K.-P., & Hua, X.-S. (2010). Real-time large scale near-duplicate web video retrieval. In A. D. Bimbo, S.-F. Chang, & A. W. M. Smeulders (Eds.), *Proceedings of the 18th international conference on multimedia 2010, firenze, italy, october 25-29, 2010* (pp. 531–540). ACM. Retrieved from `http://dl.acm.org/citation.cfm?id=1873951`

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Song, J., Yang, Y., Huang, Z., Shen, H. T., & Hong, R. (2011). Multiple feature hashing for real-time large scale near-duplicate video retrieval. In K. S. Candan, S. Panchanathan, B. Prabhakaran, H. Sundaram, W. chi Feng, & N. Sebe (Eds.), *Proceedings of the 19th international conference on multimedia 2011, scottsdale, AZ, USA, november 28 - december 1, 2011* (pp. 423–432). ACM. Retrieved from `http://doi.acm.org/10.1145/2072298.2072354`

Song, J., Yang, Y., Huang, Z., Shen, H. T., & Luo, J. (2013). Effective multiple feature hashing for large-scale near-duplicate video retrieval. *IEEE Trans. Multimedia*, *15*(8), 1997–2008. Retrieved from `http://dx.doi.org/10.1109/TMM.2013.2271746`

Szegedy, C., 0015, W. L., Jia, Y., Sermanet, P., Reed, S. E., Anguelov, D., ... Rabinovich, A. (2015). Going deeper with convolutions. In *Cvpr* (pp. 1–9). IEEE Computer Society. Retrieved from `http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7293313`

Theano Development Team. (2016, May). Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, *abs/1605.02688*. Retrieved from `http://arxiv.org/abs/1605.02688`

Usbeck, R., Ngomo, A. N., Röder, M., Gerber, D., Coelho, S. A., Auer, S., & Both, A. (2014). AGDISTIS - agnostic disambiguation of named entities using linked open data. In T. Schaub, G. Friedrich, & B. O'Sullivan (Eds.), *ECAI 2014 - 21st european conference on artificial intelligence, 18-22 august 2014, prague, czech republic - including prestigious applications of intelligent systems (PAIS 2014)* (Vol. 263, pp. 1113–1114). IOS Press. Retrieved from `https://doi.org/10.3233/978-1-61499-419-0-1113` doi: 10.3233/978-1-61499-419-0-1113

Usbeck, R., Röder, M., Ngomo, A. N., Baron, C., Both, A., Brümmer, M., ... Wesemann, L. (2015). GERBIL: General entity annotator benchmarking framework. In *Proceedings of the 24th international conference on world wide web, WWW 2015* (pp. 1133–1143). Retrieved from `http://doi.acm.org/10.1145/2736277.2741626` doi: 10.1145/2736277.2741626

Vrandecic, D., & Krötzsch, M. (2014). Wikidata: a free collaborative knowledgebase. *Commun. ACM*, *57*(10), 78–85. Retrieved from `http://doi.acm.org/10.1145/2629489` doi: 10.1145/2629489

Wang, J., Song, Y., Leung, T., Rosenberg, C., Wang, J., Philbin, J., ... Wu, Y. (2014). Learning fine-grained image similarity with deep ranking. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 1386–1393).

Wu, P., Hoi, S. C., Xia, H., Zhao, P., Wang, D., & Miao, C. (2013). Online multimodal deep similarity learning with application to image retrieval. In *Proceedings of the 21st acm international conference on multimedia* (pp. 153–162).

Wu, X., Hauptmann, A. G., & Ngo, C.-W. (2007). Practical elimination of near-duplicates from web video search. In R. Lienhart, A. R. Prasad, A. Hanjalic, S. Choi, B. P. Bailey, & N. Sebe (Eds.), *Proceedings of the 15th international conference on multimedia 2007, augsburg, germany, september 24-29, 2007* (pp. 218–227). ACM. Retrieved from `http://dl.acm.org/citation.cfm?id=1291233`

Yang, L. (2006). Distance metric learning: A comprehensive survey.

Zhang, J. R., Ren, J. Y., Chang, F., Wood, T. L., & Kender, J. R. (2012). Fast near-duplicate video retrieval via motion time series matching. In *2012 ieee international conference on multimedia and expo* (pp. 842–847).

Zheng, W.-S., Gong, S., & Xiang, T. (2011). Person re-identification by probabilistic relative distance comparison. In *Computer vision and pattern recognition (cvpr), 2011 ieee conference on* (pp. 649–656).