

Project Acronym: STAR
Grant Agreement number: 956573 (H2020-ICT-2020-1 – Research and Innovation Action)
Project Full Title: Safe and Trusted Human Centric Artificial Intelligence in Future Manufacturing Lines
Project Coordinator: INTRASOFT International



Funded by the Horizon 2020
Framework Programme of the
European Union

DELIVERABLE

D4.4 – Active Learning Systems and Techniques Initial Version

Dissemination level	PU -Public
Type of Document	Report
Contractual date of delivery	31/03/2022
Deliverable Leader	JSI
Status - version, date	Final – v1.0, 31/03/2022
WP / Task responsible	WP4
Keywords:	Active learning

Executive Summary

This document provides an overview of the activities and the results achieved in WP4 Safe, Transparent and Reliable Human-Robot Collaboration within Task 4.3 Human-Robot Interactions for Active Learning.

The main activities carried out during T4.3 are:

- 1) research in active learning, in order to facilitate human-robot interactions and accelerating knowledge acquisition;
- 2) developing prototype systems that enable human-robot interactions;
- 3) implementing active learning techniques within STAR use cases;
- 4) implementing NLP techniques and conversational clients.

The first prototypes for Task 4.3 have already been developed and are described in detail in this deliverable.

The WP4 partners conducted an extensive scientific work presented in the journal and conference publications.

Improvements to the work presented in this deliverable are expected in the next period and they will be presented in D4.5 Active Learning Systems and Techniques – Final version.

The improvements will include:

- further research and experiments on the benefits of the active learning approach,
- exploitation of the developed techniques in other STAR use cases,
- assessment of feasibility to use the developed technology in other external use cases and integration into existing tools (e.g., QlectorLEAP),
- improvement of the NLP techniques in the industrial environment,
- connection of the developed AL approaches to the knowledge base population and with it optimization of the operators time and workload.

Deliverable Leaders:	JSI: Patrik Zajec, Jože Rožanec, Klemen Kenda, Inna Novalija
Contributors:	R2M: Rubén Alonso
Reviewers:	UBI, GFT
Approved by:	Charalampos Ipektsidis, John Soldatos (INTRA)

Document History			
Version	Date	Contributor(s)	Description
V0.1	01/01/2022	JSI	Table of contents
V0.2	01/03/2022	R2M, JSI	First draft (NLP functionalities, Prototypes)
V0.3	08/03/2022	JSI	Merged the first draft
V0.4	17/03/2022	JSI, R2M	Contributions of partners
V0.5	18/03/2022	JSI	Merged second draft
V0.6	22/03/2022	JSI	Third draft
V0.7	22/03/2022	JSI	Consolidated draft
V1.0	31/03/2022	INTRA	Final QA'ed version

Table of Contents

EXECUTIVE SUMMARY	2
TABLE OF CONTENTS.....	4
TABLE OF FIGURES.....	5
LIST OF TABLES.....	6
DEFINITIONS, ACRONYMS AND ABBREVIATIONS	7
1 INTRODUCTION.....	8
2 OVERVIEW OF RELATED INITIATIVES	10
3 SYSTEM ARCHITECTURE	11
4 PROTOTYPE IMPLEMENTATION AND APPLICATION TO STAR USE CASES.	14
4.1 LOGISTICS DEMO.....	14
4.1.1 <i>Task description.....</i>	<i>14</i>
4.1.2 <i>Data</i>	<i>15</i>
4.1.3 <i>Prototype system and components.....</i>	<i>16</i>
4.1.4 <i>System Evaluation.....</i>	<i>20</i>
4.1.5 <i>Active Learning Evaluation</i>	<i>21</i>
4.1.6 <i>Conclusions</i>	<i>25</i>
4.2 AL ON IMAGES	25
4.2.1 <i>Task description.....</i>	<i>26</i>
4.2.2 <i>Data</i>	<i>26</i>
4.2.3 <i>Visual quality inspection pipeline.....</i>	<i>27</i>
4.2.4 <i>Experiments and results.....</i>	<i>30</i>
4.2.5 <i>Conclusions</i>	<i>31</i>
5 NLP FUNCTIONALITIES	32
5.1 SPEECH SYNTHESIS AND RECOGNITION APPROACHES	32
5.1.1 <i>Web Speech API for speech synthesis and recognition.....</i>	<i>32</i>
5.1.2 <i>Cloud Speech synthesis and recognition.....</i>	<i>34</i>
5.1.3 <i>Architecture.....</i>	<i>35</i>
5.1.4 <i>Proof of concept.....</i>	<i>36</i>
5.1.5 <i>Next steps and sentiment analysis.....</i>	<i>37</i>
6 CONCLUSIONS.....	38
REFERENCES	39

Table of Figures

FIGURE 1: TIMELINE OF STAR TASK 4.3..... 8

FIGURE 2: SUPPORT ARCHITECTURE FOR ACTIVE LEARNING SYSTEMS AND TECHNIQUES11

FIGURE 3: MATERIAL/CLIENT DEMANDS.....14

FIGURE 4: (A) DISPLAYS A DIAGRAM OF THE SYSTEM COMPONENTS AND THEIR INTERACTION. (B) SHOWS THE MAIN ONTOLOGY CONCEPTS WE CONSIDERED, AND THEIR RELATIONSHIPS.16

FIGURE 5: USER INTERFACE20

FIGURE 6: FLUXOGRAM, SHOWING HOW ACTIVE LEARNING AND RECOMMENDATIONS ARE IMPLEMENTED FOR MEDIA NEWS EVENT ENTRIES.....22

FIGURE 7: IMAGES EXAMPLES27

FIGURE 8: PROTOTYPE VISUAL INSPECTION PIPELINE.....27

FIGURE 9: SCREENSHOT OF THE LABELLING APPLICATION WE DEVELOPED. IN THIS CASE, WE ARE PRESENTED WITH AN IMAGE CORRESPONDING TO A "DOUBLE-PRINT" DEFECT AND A HINT IMAGE OBTAINED FROM A DRAEM ANOMALY MAP.....29

FIGURE 10: NLP COMPONENT ARCHITECTURE36

FIGURE 11: WEB SPEECH API DEMO37

List of Tables

TABLE 1: TABLE OF PEER-REVIEWED AND OTHER SCIENTIFIC CONTRIBUTIONS BASED ON THE STAR TASK 4.3.	8
TABLE 2 DATASET OF MEDIA NEWS	16
TABLE 3: OVERVIEW OF THE ACTIVE LEARNING DATASETS WITH TOTAL NUMBER OF INSTANCES PER EACH SPLIT.	21
TABLE 4: ACTIVE LEARNING AND RECOMMENDATION STRATEGIES	23
TABLE 5: LABELING QUALITY AND TIME MEASUREMENTS AND THE NUMBER OF DEFECTS CASES WHERE THE USERS COULD NOT TELL IF THERE WAS A DEFECT OR CONSIDERED THE DEFECT DID NOT CORRESPOND TO EXISTING CLASSES. BEST RESULTS ARE BOLDED, SECOND-BEST RESULTS ARE HIGHLIGHTED IN ITALICS.	30
TABLE 6: AN EXAMPLE OF JSGF GRAMMAR	33
TABLE 7: EXAMPLE OF CALCULATING THE DISTANCE BETWEEN TWO STRINGS.....	34
TABLE 8: TEXT-TO-SPEECH (CLOUD)	35
TABLE 9: SPEECH TO TEXT (CLOUD)	35

Definitions, Acronyms and Abbreviations

Acronym/ Abbreviation	Title
AI	Artificial Intelligence
AL	Active Learning
BDVA	Big Data Value Association
BoW	Bag of Words
ER	Event Registry
ERP	Enterprise Resource Planning
GAN	Generative Adversarial Networks
IIRA	Industrial Internet Reference Architecture
IISF	The Industrial Internet Security Framework
JSGF	Java Speech Grammar Format
KB	Knowledge Base
KG	Knowledge Graph
LR	Logistic Regression
MAP	Mean Average Precision
MES	Manufacturing Execution Systems
MLP	Multilayer Perceptron
NLP	Natural Language Processing
PA	Passive-Aggressive Classifier
RF	Random Forest
ROC AUC	Receiver Operating Characteristic, Area Under Curve
SA	Sentiment Analysis
SOTA	State of the Art
SR	Simulated Reality
STT	Speech-To-Text
SVM	Support Vector Machine
TF-IDF	Term Frequency - Inverse Document Frequency
TTS	Text-to-Speech
WP	Work Package
XAI	Explainable Artificial Intelligence

1 Introduction

Task 4.3 of the STAR project targets the following STAR’s WP4 Objective:

- (iii) To research and implement active learning techniques that boost human robot collaboration and accelerate knowledge acquisition.

Task 4.3 is strongly coupled with STAR Task 4.4 (Knowledge Base for Human Robot Interactions); however, it connects also STAR Task 4.1 (Explainable AI) and STAR Task 4.2 (Simulated Reality). The final products from this task, therefore, integrate functionalities (including Natural Language Processing (NLP)) from the whole work package, which is most evident in the description of the prototype of Active Learning on images described in Subsection 4.2.

This deliverable sum-ups the activities carried out in the first 15 months of the project, whereas the task has started in month 4 (M4) of the project (Figure 1). The deliverable at-hand is the initial version of the Active Learning Systems and Techniques deliverable and the final version will be delivered in M27.

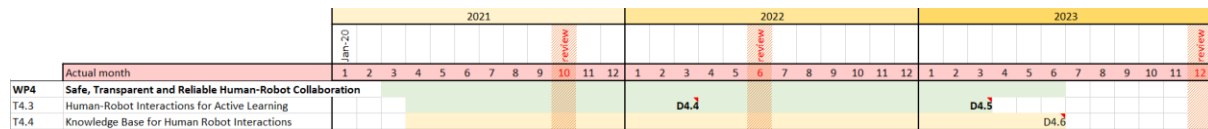


Figure 1: Timeline of STAR Task 4.3.

The work on the active learning has begun very early in the project, when data from various use cases were not yet available. In order to proceed with the work, an artificial “logistics” use case, based on Qlector’s experience and development with concrete clients in the manufacturing sector, was created. Through this use case, several methodologies have been tested and developed and were later re-used in the actual STAR use cases. The results of this initial work are described in more detail in Subsection 4.1.

Table 1: Table of peer-reviewed and other scientific contributions based on the STAR Task 4.3.

TITLE	VENUE	YEAR	TYPE	PAPER LINK
Help me learn! Architecture and strategies to combine recommendations and active learning in manufacturing.	MDPI Information	2021	Journal	link
Active Learning for Automated Visual Inspection of Manufactured Products	SIKDD	2021	Conference	link
Knowledge Modelling and Active Learning in Manufacturing	Trusted AI in Manufacturing	2021	Book Chapter	link
Towards Active Learning Based Smart Assistant for Manufacturing	APMS	2021	Conference	link
STARdom: an architecture for trusted and secure human-centered manufacturing systems	APMS	2021	Conference	link
Streaming Machine Learning and Online Active Learning for Automated Visual Inspection.	IMMS	2022	Workshop	
An Abstraction Layer Exploiting Voice Assistant Technologies for Effective Human—Robot Interaction	MDPI Applied Sciences	2021	Journal	link

It is worth noticing that research work in the STAR Task 4.3 has been extensively disseminated through journal papers, book chapters and conference presentations. Contributions are listed in Table 1. Several publications are still in preparation.

The main achievements of the work in STAR Task 4.3 are:

- Design of the system architecture of the STAR WP4 ecosystem.
- Development of an early “logistics” prototype linking explainable AI (XAI), natural language processing (NLP), and active learning (AL).
- Development of an integrated WP4 prototype linking XAI, simulated reality (SR) and AL. The prototype significantly reduces the time needed for annotation of images in an industrial setting, reduces the mistakes with the manual annotation by reduction of labelling fatigue and additional guidelines to the annotators.
- Publication of 1 book chapter, 2 peer-reviewed journal papers and 4 contributions at conferences and workshops.

The deliverable at hand is structured in 6 sections. Section 1 provides an overview of the scope and the structure of this document. Section 2 includes a review of the related work in active learning. Section 3 introduces the overall architecture of the STAR WP4 ecosystem and its integration into the STAR platform. Section 4 describes the 2 integrated prototypes and yields the initial results. Section 5 details on the implementation of NLP functionalities and finally, section 6 provides a conclusion and an overview of the next steps.

2 Overview of Related Initiatives

In this section, we briefly describe the concept of Active Learning, and related works applying Active Learning in the context of manufacturing.

Active learning is a field of machine learning that studies how to take actions to gather the data that best realizes a learning objective. Furthermore, the underlying assumption of Active Learning is that either data is abundant or easily generated, and that the highest expense is related to getting query responses (e.g. to label the data), and therefore strategies must be devised to identify the most promising data instances, that would foster models' learning [1],[2],[3]. The promise to reduce the amount of data required to train new models, along with the increasing ease of generating high-quality synthetic data, has increased the interest in Active Learning, where synthetic data generation reduces the expenses associated with data collection, while the Active Learning approach provides feedback regarding the desired characteristics of data that benefits the model the most. Nevertheless, the adoption remains low in the context of the manufacturing industry [4],[5].

Active Learning was successfully applied in a handful of manufacturing scenarios. Van et al. [6] report using machine learning to predict the local displacement between two layers on a chip in the semi-conductor industry. Given that measuring and controlling overlay errors is essential to this manufacturing process, and that such measurements are difficult and costly, active learning methods are used to select which wafers to measure, and then use them to develop predictive models that could complement such measurements. Dai et al. [7] applied machine learning to automate the optical inspection of printed circuit boards, searching for soldering defects. Machine learning algorithms require labelled data to achieve high accuracy in defect detection. Given the high cost associated with the creation of a database with enough labelled data samples, the authors applied active learning to prioritize which samples to label, and incrementally grow the database while improving the model's performance. Yue et al. [8] applied Active Learning to Gaussian Processes with uncertainties, considering the variance of the data samples related to the uncertainties associated to measurement errors and intrinsic input noise from the experimental data, to realize a better prediction performance when applied to predictive modelling for automatic shape control of composite fuselage. Finally, Taylor et al. [9] describes multiple cases where Active Learning was applied in the context of robotics, addressing issues such as prioritized decision-making, next-best-view problems, or environment modelling.

3 System Architecture

In this section, we introduce a system architecture we developed to support Active Learning systems along with other core functionalities relevant to the Work Package 4 of the STAR project. The architecture was developed keeping in mind the Big Data Value Association (BDVA) reference architecture, to ensure it can be ported to a wide variety of use cases and maximize compatibility with other big data initiatives. We propose a modular architecture to foster the use of artificial intelligence and adopt cybersecurity best practices to ensure safety, trust, and human-centric services. The architecture serves predictions, insights regarding the inner processes of the artificial intelligence models, incorporates domain knowledge and context for each prediction where needed, and helps the users by providing them with decision-making options to speed up the decision-making process. Cyber-security is applied horizontally to all the architecture modules.

We depict the components and their interactions in Figure 2: Support Architecture for Active Learning Systems and Techniques, and we provide a detailed description of the architecture modules below.

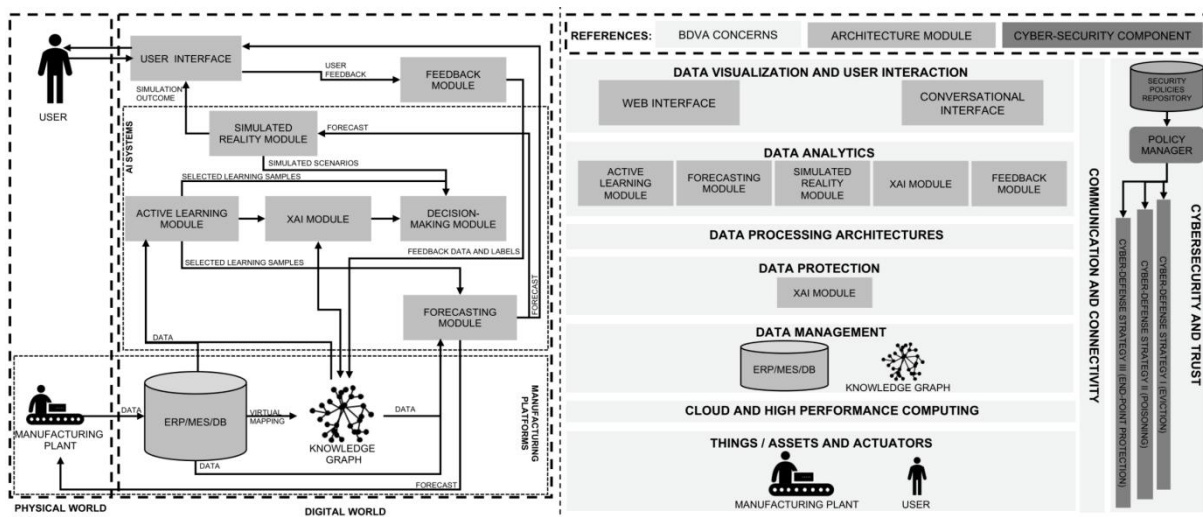


Figure 2: Support Architecture for Active Learning Systems and Techniques

The Figure¹ shows two views of the architecture we designed to support Active Learning systems and techniques. Fig. 2A (left) shows the separation between the physical and digital world, with data ingested and obtained from manufacturing platforms, while the applications are built on top of them, with particular emphasis on artificial intelligence systems. Fig. 2B (right) shows how the architecture is structured based on the BDVA reference architecture concerns, to ensure it can be ported to a wide variety of cases.

¹ The Figure was taken from: Rožanec, Jože M., et al. "STARdom: an architecture for trusted and secure human-centered manufacturing systems." IFIP International Conference on Advances in Production Management Systems. Springer, Cham, 2021.

The proposed architecture takes into account the following components:

- **storage:** stores relevant data collected either through Manufacturing Execution Systems (MES), Enterprise Resource Planning (ERP) software, and other systems. The data can be stored in databases or filesystems. Knowledge Graphs are created through virtual mapping procedures, mapping acquired data to a graph structure using a reference ontology.
- **forecasting module:** provides forecasts based on artificial intelligence and simulation models. Different models are developed, based on the goals they aim to solve (e.g., classification, regression, clustering, or ranking). Most of the models require either past data to learn patterns, or some parameterization, to predict future outcomes based on them.
- **XAI module:** takes into account forecasting models and their predictions to provide explanations to the users to better understand the inner workings of the forecasting models, and the models' rationale behind the forecasts. Explanations must be tailored to their audience, adopting a different language, level of detail, and type (e.g., feature ranking, counterfactual explanations, or contrastive explanations). Furthermore, care must be taken to avoid exposing sensitive information to stakeholders who should not have access to it. Finally, explanations can be enriched with complementary information and domain knowledge, so that the additional context can lead to better understanding, evaluation of the forecasts, and therefore responsible decision-making.
- **decision-making module:** considers inputs from inductive models and simulations, along with expert knowledge. Envisioned as a recommender system, can be implemented through heuristics or machine learning approaches to advise the users on the best decision-making options available in a given situation, and ensure the decisions lead to desired outcomes.
- **simulated reality module:** uses heuristics and models to either create alternative scenarios or generate synthetic data. These are used by (i) Reinforcement Learning algorithms to foster their learning without the need of a physical environment, or (ii) machine learning algorithms, which can leverage such simulations or synthetic data to avoid expensive data collection or mitigate the lack of data. Furthermore, simulations can be used to project possible outcomes given the users' decisions, and therefore better inform the decision-making process where possible.
- **active learning module:** uses different strategies to assess which data instances can be most important for a certain artificial intelligence model to enhance their learning, and ask human experts for input to later use them to train and enhance the model. Collecting such information can be seen as a particular responsibility of the **feedback module**.
- **feedback module:** collects feedback from users, ranging from the feedback regarding forecasts, the explanations provided by the XAI module, or decision-making options recommended to the users. Feedback can be explicit (the user provides a rating or opinion), or implicit (the omission of feedback can provide a certain signal).
- **user interface:** enables users' interactions with the system. Multimodal interactions are encouraged where necessary, complementing the use of voice with the display of relevant information on screens.

Cybersecurity frameworks and best practices traverse the whole architecture. The Industrial Internet Security Framework (IISF) provides a guide to understand and implement cybersecurity requirements as specified by the Industrial Internet Reference Architecture. It provides guidelines to secure individual components, and how such components can be bound together into a trustworthy system. The guidelines include data protection, the protection of communication and connectivity capabilities, and everything related to security configurations, management, monitoring and analytics.

The proposed architecture was partially validated through several use cases, described in "Section 4 - Prototype Implementation and application to STAR use cases".

4 Prototype Implementation and application to STAR use cases.

4.1 Logistics Demo

This prototype demonstrates a conceptual design and a developed system that guides a user from a forecast generated by a machine learning model through a sequence of decision-making steps. The system implements the architecture and components described in Section 3.

It is demonstrated on a manufacturing demand forecasting use case, providing recommendations for transport scheduling, and integrates demand forecasting models, explainable AI (XAI), a decision-making recommender system, and a knowledge graph. In addition, the system provides the means for knowledge acquisition by gathering data from users.

Finally, the prototype implements an active learning module for media events recommendation and compares nine active learning strategies through a set of experiments to understand how they balance learning and provide accurate media news recommendations to the user. The media news aims to provide additional context to demand forecasts and enhance judgment on decision-making.

The aforementioned components are used to develop decision-making workflows, which are displayed through an interactive user interface. The conversational interface was also developed, allowing the users to use only their voice when interacting with the system. Feedback is collected from users regarding forecasts, forecast explanations, and decision-making options displayed to the users. Further, labels are collected by the active learning module and used to improve the recommendation quality.

4.1.1 Task description

Supported by real data, we consider a demand forecasting use case, where the daily forecasts, issued by the AI model, are available on a material and client level (Figure 3²).

Target date: [2021-03-15](#) Material: [Material 1](#)

#	Client	Demand
1	Client 1	580
2	Client 2	2420
3	Client 3	1396

Figure 3: Material/Client Demands

Based on the forecasted demand, the user schedules the transports to deliver the material to the specific clients. Scheduling is performed through a sequence of non-trivial decision-making steps that guide the user from the forecast to the final logistics plan. At each step, the user selects one of the possible decision-making options. For example, in the first step,

² The Figure was taken from: Zajec, P.; Rožanec, J.M.; Trajkova, E.; Novalija, I.; Kenda, K.; Fortuna, B.; Mladenčić, D. Help Me Learn! Architecture and Strategies to Combine Recommendations and Active Learning in Manufacturing. Information 2021, 12, 473.

the user decides if the particular material should be shipped with an existing transport or new transport should be scheduled, while in the second step, the user selects the type of the transport and the departure date from the possible options, etc.

Along the way, there are multiple opportunities for the user to provide feedback back to the system and, in such a way, help the system to better capture the knowledge about the use case and possibly improve its capabilities over time. The feedback can be targeted towards the forecast itself or the feedback can be given during the decision-making steps, for example, so that the model better understands why a particular decision-making option was selected.

To enhance the user's awareness in a demand forecasting setting, the stream of media news on events that can influence demand is available. For example, when forecasting engine parts for the automotive industry, such stream includes the news regarding the automotive industry, the global economy, unemployment, and logistics. While media news can be retrieved from some news intelligence platforms, keywords based queries can issue many false positives. It is thus imperative to develop a recommender system capable of discriminating and prioritizing good quality news over those considered false positives. Furthermore, it is desired that such a recommender system improves the quality of discrimination over time and requires as little manual labelling effort as possible.

To summarize, the task is to design and implement a system that would support the described decision-making flows while collecting feedback from the users along the way. Note that such a setting, where we start with some forecasts and make plans based on sequential decision-making, is common to multiple tasks in manufacturing. Methodology and components developed when solving this particular task are therefore expected to be more generally applicable.

4.1.2 Data

We have used the dataset from a European automotive original equipment manufacturer, containing three years of daily demand corresponding to 279 materials and 149 clients. The regression model was trained and evaluated for all the clients and materials, while we only used a subset to evaluate our system.

For each forecast, we have generated the forecast explanations. Such explanation provides an additional insight into the rationale behind each forecasted quantity, which reveals the inner working of the AI model. Explanations are relevant for our task as the users can provide feedback on them and thus help to improve the system.

Next, we tried to make the transport scheduling process realistic. There was no data regarding the physical characteristics of each product/material and also no information regarding the specific addresses of the clients ordering such products, except the destination country. To mitigate these constraints, we collected pricing and delivery time information for air, land, and sea shipments considering single standard forty feet containers from Slovenia to fourteen countries. Such data was retrieved from two specialized web-pages (we collected data regarding pricing and shipment time from World Freight Rates³ and SeaRates⁴).

³ <https://worldfreightrates.com/freight>

⁴ <https://www.searates.com/freight>

We have also collected the dataset of media news events classified into four categories: Automotive Industry, Global Economy, Unemployment, and Logistics. The dataset was manually annotated by three human annotators, based on the specific keywords used in each category to retrieve them (see Table 2). The media news events were retrieved daily for a period of six months (from July 2019 to December 2019) from Event Registry [10], a well-established media events monitoring platform that has monitored mainstream media since 2014.

CATEGORY	KEYWORDS	# INSTANCES	MISSING DATA	MEPD
(A) Automotive Industry	car sales demand, new car sales, vehicle sales, car demand, automotive industry	3865	10 days	20
(B) Global Economy	global GDP projection, global economic outlook, economic forecast	853	29 days	5
(C) Unemployment	unemployment rate, unemployment numbers, unemployment report, employment growth, long-term unemployment	3801	8 days	22
(D) Logistics	logistics, maritime transport, railroad transport, freight, cargo transport, supply chain	28,231	0 days	133

Table 2 Dataset of Media News

4.1.3 Prototype system and components

When realizing a system suitable for our task, we have followed the architecture and components presented in Section 3. Further, we have also developed a methodology to identify relevant decision-making options, information and feedback of interest, and how to collect them. We provide the description of our components (see also Figure 4⁵) below and illustrate, how they relate to the task.

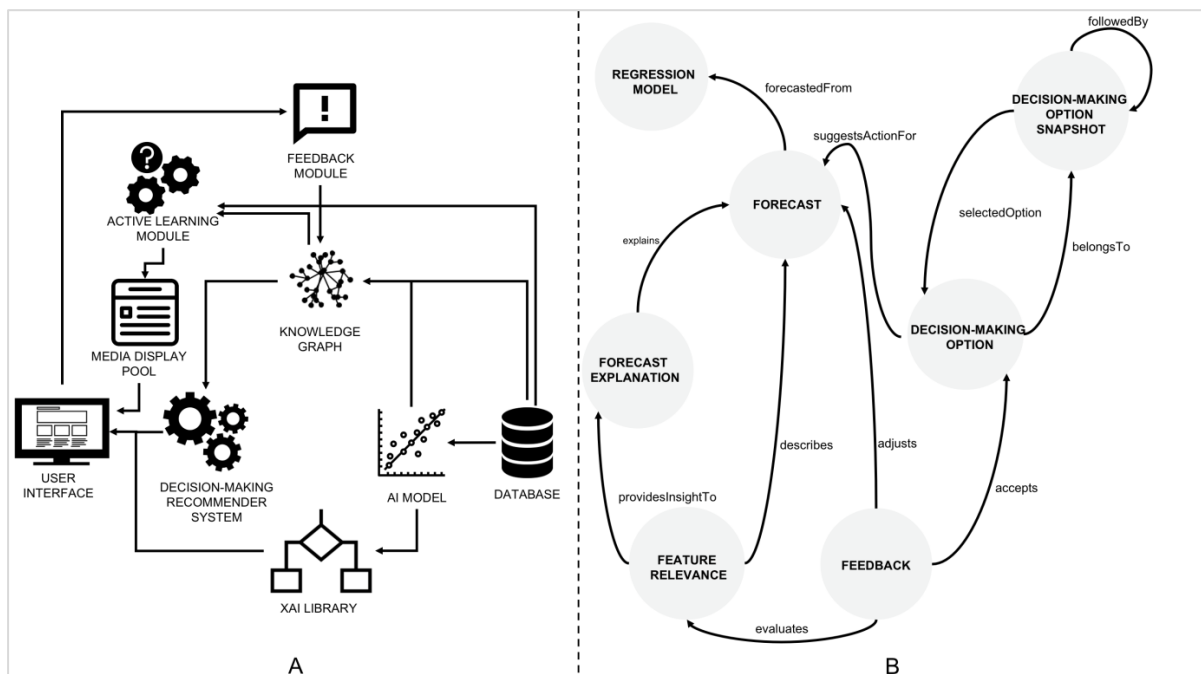


Figure 4: (A) displays a diagram of the system components and their interaction. (B) shows the main ontology concepts we considered, and their relationships.

⁵ The Figure was taken from: Zajec, P.; Rožanec, J.M.; Trajkova, E.; Novalija, I.; Kenda, K.; Fortuna, B.; Mladenčić, D. Help Me Learn! Architecture and Strategies to Combine Recommendations and Active Learning in Manufacturing. Information 2021, 12, 473.

DATABASE

Stores operational data from the manufacturing plant. Data can be obtained from ERP, MES, or other manufacturing platforms.

AI MODEL

Aims to solve a specific task relevant to the use case, such as classification, regression, clustering, or ranking. In our prototype, the AI model was developed to forecast the demand on a material and client level daily.

XAI LIBRARY

Generates the forecast explanations for the forecasts from an AI model. We generated forecast explanations using the LIME library [11], but other approaches could be used too (e.g., Lion Forests [12] or Shapley values [13]).

KNOWLEDGE GRAPH

The knowledge graph is a central component of the system. Instantiated from an ontology, it relates forecasts, forecast explanations, decision-making options, and feedback provided by the users. To ensure context regarding decision-making options and feedback provided is preserved, different relationships are established. We have developed a more general ontology suitable for all use cases that have a similar setting as our task. Each use case extends the ontology with its own concepts and relations while the general structure remains the same.

The main ontology concepts and relations can be seen in Figure 4. The feedback entity directly relates to a forecast, forecast explanation, and decision-making option. While a chain of decisions can exist for a given forecast, there is a need to model the decision-making options available at each stage and the sequence in which they are displayed. To that end, the decision-making snapshot entity aims to capture a list of decision-making options provided at a given point in time. A relationship between decision-making option snapshots (followedBy) provides information on such a sequence. For each decision-making snapshot, a selectedOption relationship is created for the user's selected decision-making option. To link the first decision-making options to the forecast, the *suggestsActionFor* relationship is created between the forecast entity and entities that correspond to the first decision-making options displayed for that forecast. Since the decision-making options are linked to the decision-making option snapshot and preserve a sequential relationship, all decision-making options can be traced back to the forecast that originated them.

DECISION-MAKING RECOMMENDER SYSTEM

Recommends decision-making options to the users. Recommended decision-making options can vary depending on the users' profile, specific use case context, and feedback provided in the past.

Demand forecasts influence decision-making in a wide variety of scenarios: from raw material orders to workers hiring and upskilling to logistics arrangements to meet the required deadlines. Decision-making recommender systems can alleviate such decision-making by suggesting to the user appropriate actions based on the projected demand. We consider two possible scenarios. The first scenario refers to the user who schedules a new transport for a given demand, material, client, and date. Here, the decision-making options are the possible transports, differing in transport type, delivery time, and price. The second

possible scenario relates to the user who decides to use an existing transport. Here each decision-making option selects one of the existing transports. In both steps, the recommendation module ranks the decision-making options from most to least relevant. We developed two recommendation strategies:

- The *heuristic-based* recommender system follows simple rules, hand-crafted either by the domain expert or simply by the system's developer based on his incomplete knowledge about the problem. Such a system has no learning capacity, and therefore has little potential to improve the users' experience. The recommendation quality directly depends on the quality of the designed rules. An example of such a heuristic rule is consistently ranking the transports according to the price or keeping only existing transports delivering in the client's proximity and ranking them according to the remaining capacity.
- The *knowledge-based* approach provides recommendations based on the feature vectors' similarity to a target vector describing users' requirements. To that end, each decision-making option at the given step is represented as a vector. The representation captures all necessary information for the ranking, encoding the context up to the current step, the corresponding decision-making option, and its relation to all other possible decision-making options (the decision-making options snapshot). The model assigns the relevance score to each option. The ranking is determined by sorting the scores from highest to lowest. The representation and the underlying model should be expressive enough to cover the scenarios encountered in the use case. As with the heuristic-based strategy, domain knowledge heavily influences the design of features, but the content-based strategy provides greater flexibility. The features directly capture the context, which in our use case includes the forecasted demand, date, material, and client; the decision-making option, which in the case of scheduling a new transport includes the transport type, time of delivery, capacity, and price; and the relation of the decision-making option to the whole set of available options to capture, how this option is different from others and why it should be preferred.

ACTIVE LEARNING MODULE

Aims to select data instances whose labels are expected to be most informative to a machine learning model and thus are expected to contribute most to its performance increase when added to the existing dataset. Obtained labels are persisted to the knowledge graph and database.

Focus of the active learning module is media news recommendation. We built a set of active learning binary classifiers (4 to be exact, as there are 4 media news categories in our dataset), each one informing if the media news considered does fit into a specific media news category or not. We consider the end-user is at the same time the news consumer and the active learning's *oracle*, providing feedback regarding unlabelled instances. In our design, we display the news and collect feedback regarding them in the same user interface. This poses an exploration-exploitation dilemma since the same user interface space must be optimized to provide high-quality media news balancing between those entries where high confidence in the category exists but provide little additional information to the existing dataset, and those entries where the confidence is lower, but can provide a higher degree of novelty to the dataset.

MEDIA DISPLAY POOL

Contains the media news selected by the active learning module, which are to be displayed to the user. In particular, on each day, at most ten pieces of news per each of the four categories are displayed to the user. The user can then provide positive or negative feedback (label) regarding each piece of news. The news should be informative for the system as the goal is to achieve good classification performance as soon as possible. On the other hand, the displayed news events should also be relevant so that the system is usable to the users after the first few iterations.

FEEDBACK MODULE

Collects feedback from the users and persists it in the knowledge graph. The system collects feedback regarding forecasts, the explanations provided by the XAI module, or decision-making options recommended to the users. Feedback can be explicit, for example, by editing the forecast value or marking the features from the forecast explanation as non-meaningful, or implicit, for example, by not editing the forecast value which could imply that the forecast seems correct or by choosing one of the given decision-making options which implies that the option is most meaningful in the given context.

USER INTERFACE

To provide forecasts, forecast explanations, media news, and decision-making options to the user, we developed a user interface with five distinct parts (see Figure 5⁶). Among them we find:

A) **Media news panel:** displays media news regarding the automotive industry, global economy, unemployment, and logistics. The user can provide explicit feedback on them (if they are suitable or not), acting as an oracle for the active learning classifier. Once feedback is provided, a new piece of news is displayed to the user.

B) **Forecast panel:** given the date and material, it displays the forecasted demand for different clients. For each forecast, three options are available: edit the forecast (providing explicit feedback on the forecast value), display the forecast explanation, and display the decision-making options. The lack of editing on displayed forecasts is considered implicit feedback approving the forecasted demand quantities.

C) **Forecast explanation panel:** displays the forecast explanation for a given forecast. Our implementation displays the top three features identified by the LIME algorithm [11] as relevant to the selected forecast. If users consider that some of the displayed features do not explain the given forecast, they can provide feedback by removing it from the list.

D) **Decision-making options panel:** displays possible decision-making options for a given forecast or step in the decision-making process. In particular, the decision-making options relate to possible shipments. If no good option exists, the user can create its own.

⁶ The Figure was taken from: Zajec, P.; Rožanec, J.M.; Trajkova, E.; Novalija, I.; Kenda, K.; Fortuna, B.; Mladenić, D. Help Me Learn! Architecture and Strategies to Combine Recommendations and Active Learning in Manufacturing. *Information* 2021, 12, 473.

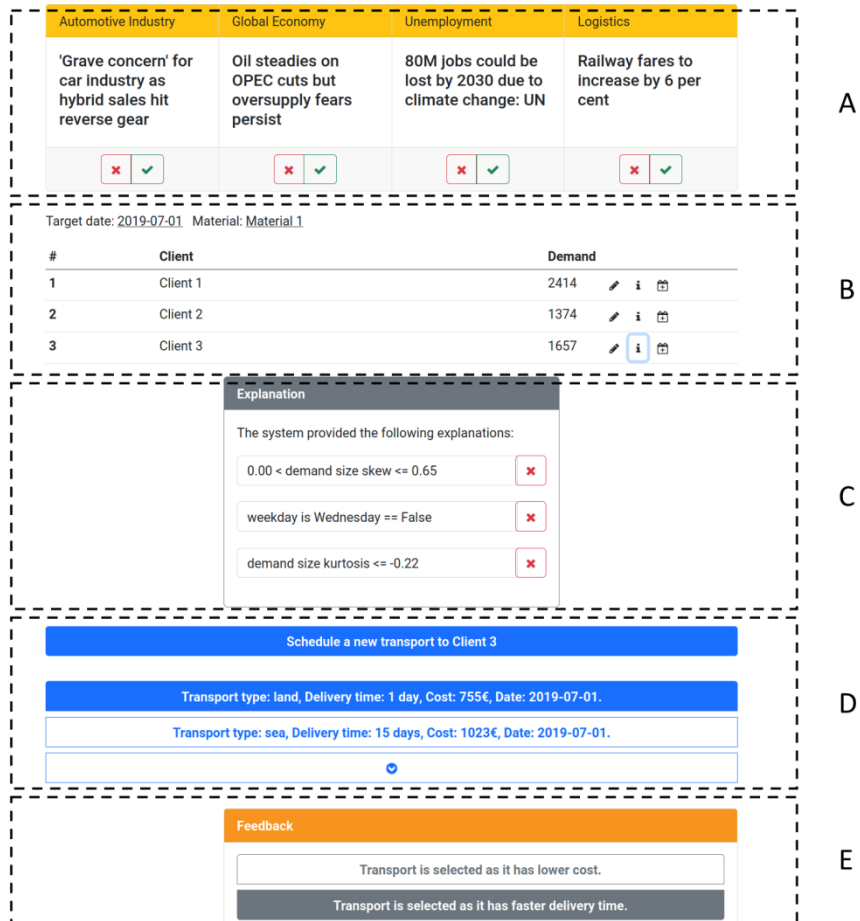


Figure 5: User Interface

E) **Feedback panel:** gathers feedback from the user to understand the reasons behind the chosen decision-making option. While some pre-defined are shown to the user we always include the user’s possibility to add their reasons and enrich the existing knowledge base. Furthermore, such data can be used to expand feedback options displayed to the users in the future.

CONVERSATIONAL INTERFACE

Supports voice-only interactions with the system. We have integrated the speech-to-text module presented in Section 5 to the user interface. This allows the user to interact with the system using a set of pre-defined voice commands.

4.1.4 System Evaluation

To evaluate the proposed architecture from the Section 3, empirical evaluations and the development of a concrete use case were utilized. In particular, we applied it to develop a system that enables displaying forecasts, forecast explanations, and decision-making options, provide a decision-making workflow, and means to collect feedback and knowledge from the users⁷.

⁷ A video demonstrating the application, including the conversational interface, is available at: <https://youtu.be/EpFBNwz6KIk>

The system we developed for the particular use case of demand forecasting proves that a semantic approach enables effective and flexible means to solve complex knowledge acquisition and solve interoperability conflicts. Since the system was not deployed to production environments, we cannot assess the perceived quality of the forecast explanations, and the impact of the given forecast explanations and decision-making options provided.

Further, we did an extensive evaluation of the active learning module for media news recommendations.

4.1.5 Active Learning Evaluation

DATASET

The active learning experiments were performed on a dataset of media news events, presented in the section above. The first month of the dataset was reserved for training the initial version of the models and for tuning the model’s hyperparameters. The last month of the dataset was reserved for testing the classification performance of the models at each active learning step. The remaining data was used to execute the active learning experiments and evaluate the recommendation performance at each step.

Table 3: Overview of the active learning datasets with total number of instances per each split.

	A	B	C	D
Initialization set size (# instances)	607	128	638	4388
Learning set size (# instances)	3070	693	2882	23051
Test set size (# instances)	795	160	919	5180
Ratio of negative instances (all sets)	69.83%	65.06%	92.50%	97.71%
Ratio of positive instances (all sets)	30.17%	34.94%	7.50%	2.29%
Number of AL iterations	122	115	122	123
Number of possible selected instances (given $k = 10$)	1138	644	1132	1205

The ratio of negative and positive instances, as labelled by the human annotators, number of iterations on the learning set when AL is used and a maximum number of possible queried instances by limiting the budget with $k = 10$ instances per iteration. "# instances" denotes "the number of instances".

We provide an overview of the dataset in Table 3. We report the dataset size regarding the number of instances per dataset split (initialization set, learning set used with AL, and test set). We can observe that the datasets vary in size, with B being the smallest and D being almost two magnitudes larger than B. Further, we include the ratio of negative and positive instances, as labelled by the human annotators. The datasets are differently balanced, with D being most unbalanced as the ratio of positive instances is only 2.29%. We consider that the datasets’ diversity strengthens the experiments designed to evaluate diverse scenarios.

In addition to the dataset information, in Table 3 we also report the number of AL iterations per each dataset (the number of days when at least one news event is available) and the maximum number of instances that could be queried for a category, per day, on average. While we conducted the experiments with a fixed budget of at most k data instances per day; note that this number is smaller than k times the number of days; less than k instances were available for some days.

ACTIVE LEARNING PROCEDURE

We executed the following procedure (see Figure 5⁸). For each day, we retrieved all available events for that given day, and for each media news entry, we created the corresponding feature vector and assessed whether it should be displayed to the user to gather feedback (label the instance) or not. This decision was made based on a strategy that considered how informative the news entries were to the existing dataset, and their quality towards the target category, given the requirement that the events should be both relevant for the user (recommendation quality) and informative for the model (improvement of classification). For each day, we selected at most k events, which were then shown to the user. We set $k = 10$, based on the median number of events per day, and acknowledging it is a common practice to query a fixed number of instances at each step according to the literature. Once the media entry was displayed to the user, it was incorporated into the existing dataset if it provided an annotation.

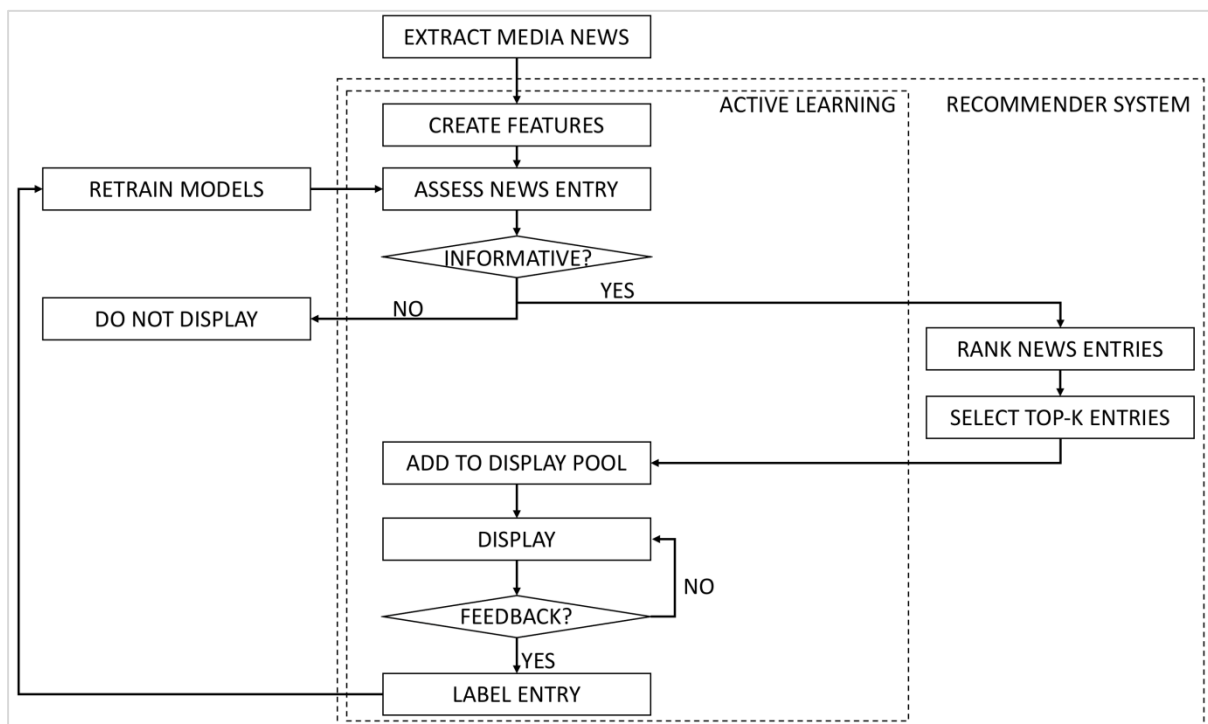


Figure 6: Fluxogram, showing how active learning and recommendations are implemented for media news event entries

INSTANCE SELECTION STRATEGIES

We have evaluated nine active learning strategies for selecting relevant events at each iteration, balancing learning and recommendation (see Table 4).

⁸ The Figure was taken from: Zajec, P.; Rožanec, J.M.; Trajkova, E.; Novalija, I.; Kenda, K.; Fortuna, B.; Mladenčić, D. Help Me Learn! Architecture and Strategies to Combine Recommendations and Active Learning in Manufacturing. Information 2021, 12, 473.

Table 4: Active learning and recommendation strategies

STRATEGY	DESCRIPTION
Random	Selects the k random instances at each step.
Uncertain	Selects k instances with highest uncertainty score at each step.
Certain	Selects k instances with lowest uncertainty score, that is, most certain examples.
Positive uncertain	Select at most k instances that were labeled as positive by the classifier and have the highest uncertainty scores.
Positive certain	Select at most k instances that were labeled as positive by the classifier and have the lowest uncertainty scores.
Positive certain and uncertain	Select at most $k/2$ positive points with lowest and at least $k/2$ points with highest uncertainty score.
Alpha trade-off ($\alpha = 0.5, 0.75, 1.0$)	We adapt the strategy proposed by [21]

Strategy *Uncertain* implements the uncertainty assumption that labels of the instances with the highest classification uncertainty are the most informative. It solely focuses on learning as such instances tend not to be the most relevant for the recommendation.

The *Random* strategy is included as a baseline, and so is the *Certain* strategy, which only selects the least uncertain instances whose labels should provide the most negligible value for the system according to the uncertainty assumption.

To also address the recommendation, the *Positive uncertain* strategy selects the instances labelled as positive by the model as this already signals that the instance is likely to be relevant for the recommendation. At the same time, it might still provide some value for learning due to uncertainty. On the other hand, the *Positive certain* strategy selects only the positive instances. Therefore, it ranks them according to the certainty, which should highly favour the recommendation while providing little value for learning.

The *Positive certain and uncertain* strategy tries to include both recommendation and learning by following the *Positive certain* strategy for the first $k/2$ instances (or less if there are not enough positive instances) to provide relevant recommendations and next following the *Uncertain* strategy to select at least the $k/2$ instances relevant for learning.

The *Alpha trade-off* strategy is adapted from [14] and has a parameter α , used to control between learning and recommendation. According to [14], $\alpha=0.5$ selects the instance with highest uncertainty from the pool and thus favours learning while $\alpha=1.0$ selects most certainly positive instance and thus favours recommendation. Setting $\alpha=0.75$ could therefore provide a trade-off between learning and recommendation.

CLASSIFICATION MODELS

Only the titles of news events were considered for classification. We used two groups of classification models, namely, the ones that are retrained on the labelled set on each iteration of AL and the ones that are trained incrementally (online) on each new presented labelled instance. We used logistic regression (LR), support vector machine (SVM), and random forest (RF) in the first group. Among the online algorithms, we used a passive-aggressive classifier (PA) [15].

We experimented with three text representation techniques: TF-IDF weighted BoW representation, which is a classical representation technique used for text classification and serves as a strong baseline in our experiments; an average of token embeddings from the

RoBERTa model (We have used the pre-trained version of “RoBERTa-base” model implemented in the Huggingface library [16]) which proved to be most effective for text classification based on the results obtained by [17], and representations obtained from the Universal sentence encoder [18].

Through our experiments, we focused on comparing a set of data selections strategies, retrieving unlabelled data from a pool of data instances. We simulated a realistic scenario, where the news events were presented on a daily level, and the model received mini-batch of labelled instances. We assume that the set of labelled instances always fit in the available memory so the batch models can be re-trained in each iteration to achieve the best performance. In addition to the batch models, we also tested several streaming models, from which the best performance was obtained with the Passive-Aggressive classifier, and the results included in this work. We consider that, given that there is no need to display media news in real-time, and that providing them at a daily granularity is enough, having a pool of news provides greater flexibility when choosing unlabelled data instances and choice of machine learning models. Nevertheless, our goal is to compare selection strategies. Thus, the models are useful for providing if a strategy is consistently better across several aspects through the models of choice.

RESULTS

Strategies and models were evaluated in the AL setting by following the procedure explained in the Section 4.1.5. The aim was to evaluate the proposed active learning strategies and their ability to tackle the learning versus recommendation trade-off, regardless of the dataset, model and representation used.

We use a separate test set to measure the active learning models’ classification performance to evaluate the models. In contrast, the recommendations’ quality is measured at each step of active learning using the gold labels of the displayed news events. To measure the models’ discrimination power, we adopted the ROC AUC, a widely used metric due to its invariance to a priori class probabilities. On the other hand, to measure the quality of the recommendations, we adopted the MAP metric, which computes the precision of the recommendation set, and is not affected by the number of entries considered in each particular case (the desired property when $k' < k$ media news events are shown to the user).

We execute multiple experiments and obtain results for each strategy, dataset, model and representation. We are focused on evaluating different aspects of the strategies, regardless of the choice of the dataset, model, and representation. Therefore, we first group the results by model, representation, and dataset. Then, inside each group, we sort and rank the strategies according to their aspect-specific score. Finally, we calculate the mean rank for each strategy and use it to order the strategies. We determine the significance of differences between the strategies using the statistical test, more specifically using Wilcoxon signed-rank test [19] at a p-value = 0.005. We compare the strategies according to their:

- Classification performance: measured by the ROC AUC score obtained on the test set in the last iteration of active learning. It measures the final performance of the classification models, trained on instances selected according to the specific strategy.
- Recommendation precision: measured by the MAP metric for all active learning iterations. MAP enables us to quantify the strategies’ performance on how accurate the recommended entries are while penalizing their ordering within the top K entries.

- Recommendation recall: measured by the recall score for all active learning iterations. Recall evaluates how many of the relevant instances were actually recommended and displayed to the user. While MAP score measures how many of the k (or less) displayed instances are relevant and whether the relevant instances are shown first, the recall score measures the ratio of shown relevant instances versus all relevant instances.

Through the classification and recommendation results, we have evaluated how well each strategy performs in terms of learning and recommendation and how its performance compares to others. As just a single strategy is implemented in the active learner, it has to be such that it best balances the learning and recommendation for the best user experience. Based on the results, we consider the *Alpha trade-off* ($\alpha= 1.0$) strategy to be the best choice, followed by the *Positive certain and uncertain* strategy. The classification results showed no statistically significant difference in performance between the best strategies. Although based on the precision of recommendation (MAP score) results, the *Positive certain* is the best performing strategy, it only performs well in one aspect of recommendation and ignores the recall. Both Alpha trade-off ($\alpha= 1.0$) and Positive certain and uncertain are second-tiers in terms of MAP score with Alpha trade-off ($\alpha= 1.0$) strategy being slightly better, while they rank first and second in terms of recommendation recall.

Additional results and detailed analysis can be found in [20].

4.1.6 Conclusions

The prototype presents an architecture designed to acquire and encapsulate complex knowledge using semantic technologies and artificial intelligence. The system was instantiated for the demand forecasting use case in the manufacturing domain, using real world data. In particular, the system provides forecasts and explanations, enriches users' domain knowledge through a set of media news, recommends decision-making options, and collects users' feedback.

Furthermore, the system uses active learning to reduce manual labelling effort and better discriminate between good and bad media news reporting events related to the demand forecast domain. A series of experiments were executed to understand the best exploration and exploitation trade-off between strategies, which is required to learn from unlabelled media news entries while providing good recommendations to the users.

We consider that the best performance was achieved by the Alpha trade-off ($\alpha= 1.0$) and Positive certain and uncertain, which displayed a strong performance in terms of MAP score and recall. It shall be noted that while many improvements can be introduced to increase the classification performance on top of the existing datasets, our research mainly focused on evaluating the impact of each strategy on learning.

4.2 AL on images

Quality control allows companies to verify the products' conformance to requirements and specifications and thus build customers satisfaction and the brand's reputation. Artificial Intelligence enables higher degrees of visual inspection automation, reducing inspection times while ensuring all products are evaluated under the same criteria.

In this prototype, we address the problem of an automated visual inspection setup complemented with a manual revision process. We envision that such a visual inspection process can leverage active learning to continuously improve the classification models' quality based on newly labelled data.

We develop and demonstrate a visual inspection pipeline to automate the visual quality inspection, reduce the amount of manual workload, assist the human annotators, and monitor their performance. We combine XAI (T4.1), GANs (T4.2) and active learning (T4.3) techniques, to enhance the manual revision process by selecting the most informative unlabelled data, hint human annotators on possible defect locations, and synthesizing images that can be intertwined into the data stream with multiple purposes, such as evaluate the annotators' attention or reduce the class imbalance.

4.2.1 Task description

The task is based on a real-world use case regarding the Philips Consumer Lifestyle BV. The company produces numerous products on which they print their logo. One of the visual inspection tasks is concerned with detecting any defective printing to ensure only products where the logo was correctly printed are delivered. The printing of the company logo is performed with different pad-printing setups. Printed products are later inspected. If a defective print is found, the product is removed from the production line.

If done manually, such inspection poses several challenges, such as limited scalability (e.g., inspectors must be trained, and such effort can grow proportionally to the production scale) and operator-to-operator inconsistency. Therefore, the development and implementation of the automated visual quality inspection models is required. Such models can reduce manual work and ensure the speed of the visual quality inspection matches the production speed.

The models require data to be trained. Supervised models require annotated data, which further requires manual labour. To best use the available annotation resources, the annotation process must be thought out well and simplify the task for the human workers as much as possible. This can be achieved, for example, by carefully selecting the instances that should be annotated and by providing different hints on the possible label of each instance.

Visual quality inspection datasets, collected from the real stream, ought to be highly imbalanced, as most of the instances tend to be of good quality, that is, they contain no defect. Without using proper balancing techniques, the models' performance can be significantly degraded, especially in the minority classes.

The identified challenges for the task are: I) development of automatic visual quality inspection models, II) ensuring the robustness of such models as their performance directly impacts the consumers satisfaction and consequently the brand's reputation, III) development of the data collection and annotation pipeline which assists the human annotators to perform the annotation task efficiently.

4.2.2 Data

Philips Consumer Lifestyle BV provided a dataset of 3.518 images. The images were labelled into one of three possible categories: good print (no defect was observed), double printing,

or interrupted printing (see Figure 7⁹). The dataset is highly imbalanced with most of the images being in the good print category.



Figure 7: Images Examples

4.2.3 Visual quality inspection pipeline

Based on data provided by Philips Consumer Lifestyle BV, we developed a prototype visual inspection pipeline (see Fig. 8), leveraging machine learning models at various stages to automate the visual inspection, reduce the amount of manual workload, assist the human annotator, and monitor their performance. The precondition to the pipeline is (i) a labelled dataset to train a supervised machine learning model (ii) a stream of incoming images to be inspected.

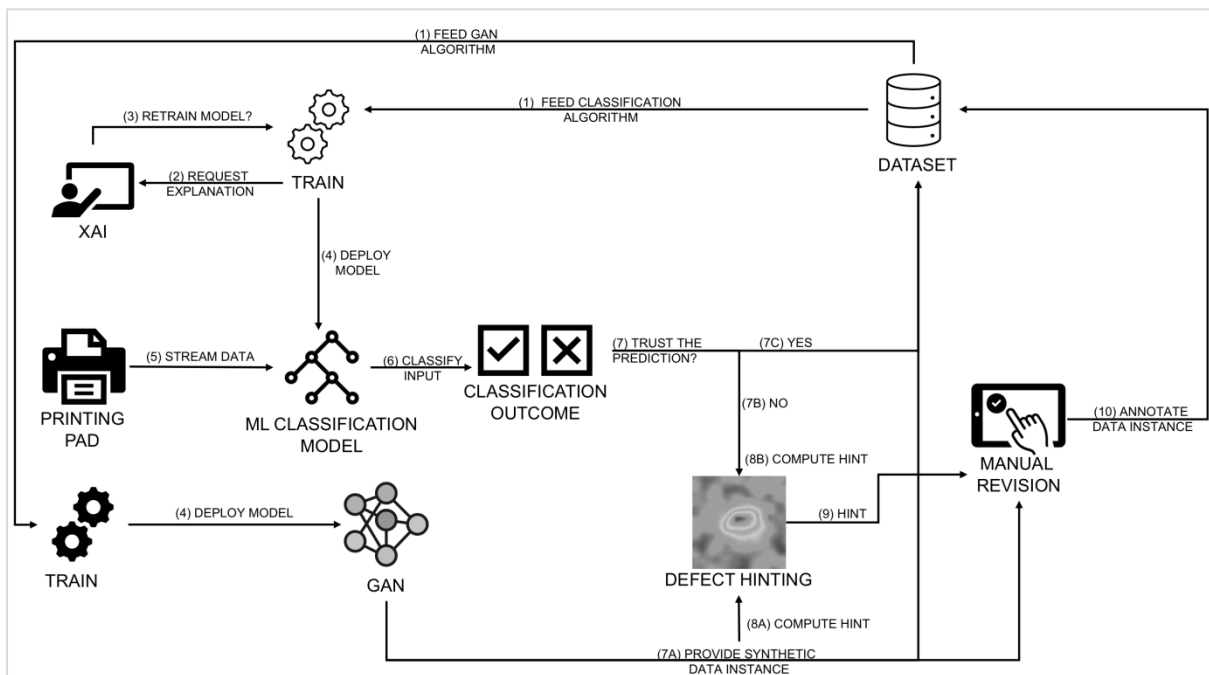


Figure 8: Prototype Visual Inspection Pipeline

While in a real-world deployment, the stream of images to be inspected originates in the printing pad, we simulate it with data from the dataset. We envision the pipeline has two sources of data: (i) the printing pad, which provides a stream of images regarding all the manufactured products to be inspected, and (ii) a Generative Adversarial Network, which

⁹ The Figure was taken from: Rožanec, J.M., Trajkova, E., Dam, P., Fortuna, B., & Mladenčić, D. (2021). Streaming Machine Learning and Online Active Learning for Automated Visual Inspection. ArXiv, abs/2110.09396.

generates synthetic images that simulate either good or defective products, and whose labels are known altogether with the generated image. We detail the interplay of both sources further in this section.

The prototype visual inspection pipeline (see Figure 8) leverages machine learning models at various stages. The precondition to the pipeline is (i) a labelled dataset to train a supervised machine learning model (ii) a stream of incoming images to be inspected.

The pipeline can be divided into the following components.

DATASET

Contains labelled images originating both from the printing pad and from the generative model. We envision that unsupervised classification models can reduce the cost of the initial dataset creation. This can be achieved by automatically labelling images that correspond to non-defective products where the classifier has enough certainty about the classification outcome while leaving the rest of the data instances to the human annotator.

We have implemented the unsupervised classification model with DRAEM [21], which is considered SOTA for unsupervised anomaly detection.

CLASSIFICATION COMPONENT

Implements the automated visual quality inspection model based on machine learning. Once the labelled dataset is ready, we use it to train a supervised machine learning classification model. The machine learning engineer evaluates the quality of the model, assessing (i) the classification metrics, (ii) insights from explainable artificial intelligence models (to understand whether the model learned correctly), and (iii) which prediction score thresholds must be used to deliver the expected quality and minimize manual revision workload. Once the model is deployed, it classifies incoming images from the printing pad.

Given the natural imbalance of the quality inspection data, most cases are labelled as non-defective pieces with enough confidence not to require any manual intervention. The pieces where there is not enough confidence regarding the predicted label could be temporally separated from the production line until a human can confirm whether the piece is defective or not. We envision all inspections to be done through an application interface devised for manual revision, where decisions are made based on the images displayed to the user.

The supervised classification model was selected according to our research [22], being the one with the highest performance in both standard supervised and active learning settings. It classifies the images into one of the three categories (no-defect, double printing or interrupted printing). It consists of a pre-trained ResNet-18 model [23] for feature extraction, extracting a 512 values long feature vector from the Average Pooling Layer, and an MLP with two dense layers (with 512 and 100 features), an intermediate ReLU activation, and a softmax activation at the end. To better understand the models' rationale behind a prediction, we have used GradCAM [24], which inspects gradient information to determine neuron activations in the last convolutional layer of a neural network.

SIMULATED REALITY COMPONENT

Implements the synthetic image generation. As mentioned above, the pipeline has two sources of data, namely the real images from the printing pad, and the synthetically generated ones. Generation of realistic synthetic images can be seen as a data augmentation technique, which helps us tackling the class imbalance in the dataset. It

further gives us the possibility to simulate the images stream while allowing us to set the ratio of defects – a feature that turns out to be useful for the annotation component.

Generative Adversarial Networks (GANs) have achieved a prominent space for image generation [25]. We use Lightweight GAN [26] model due to its superior performance both in terms of training speed and quality of generated images. The model is especially suitable when number of images is small, which is true for both double printing and interrupted printing classes. We train one model per each class.

ANNOTATION COMPONENT

For the manufactured products that undergo manual revision, we developed an application interface (see Figure 9), displaying an image of a good sample, the image that corresponds to the current piece, and eventually some defect hint.

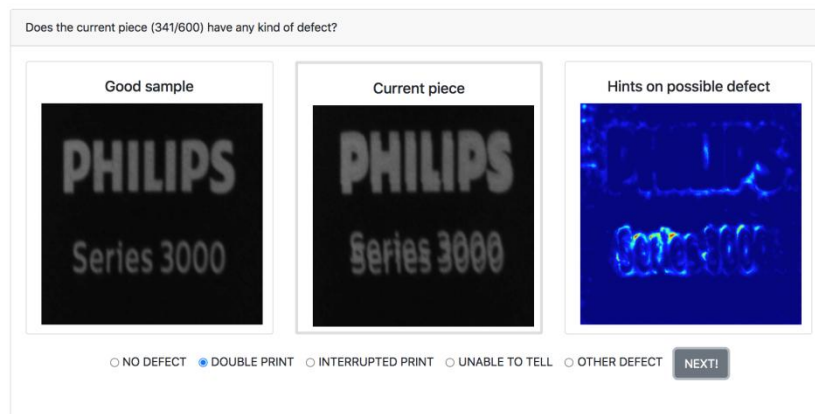


Figure 9: Screenshot of the labelling application we developed. In this case, we are presented with an image corresponding to a "double-print" defect and a hint image obtained from a DRAEM anomaly map.

Furthermore, we explored intertwining synthetic images generated by a GAN model with those that originated at the printing pad. While the label is unknown for the images that originated at the printing pad, it is not the case for the synthetic ones. Such images can therefore serve multiple purposes, such as monitoring the users' quality of labelling or preventing class imbalance in the data stream to avoid labelling inaccuracies due to users' fatigue (e.g., if a hundred images correspond to non-defective pieces, it is most likely, that the user will not notice the defective one except if the defect is easy to notice). Once a new labelled instance is obtained (either corresponding to a real or synthetic image), it is stored in the dataset to be used to develop future versions of the machine learning models.

Visualizations, such as heatmaps, allow users to understand the models' rationale behind the prediction. However, they could also be used to hint at where the defect could be in a visual inspection setting. We consider three types of hints to assist the human annotator: 1) DRAEM anomaly map for defect hinting, where DRAEM is the SOTA model for unsupervised anomaly detection; 2) GradCAM heatmap for defect hinting, based on the GradCAM explanation of the most probable class as predicted by the visual quality inspection classifier; 3) nearest labelled image from the dataset, considering the structural similarity index measure, together with its label.

4.2.4 Experiments and results

We developed two classification models and a GAN model to demonstrate the pipeline. Furthermore, we implemented an application to execute multiple manual revision experiments and better understand the users' experience and process shortcomings.

To assess the quality of the classification models, we computed the Area Under Curve of the Receiver Operating Characteristic (AUC ROC), a widely adopted classification metric. The metric is invariant to a priori class probabilities, a relevant aspect for imbalanced datasets. Finally, to evaluate how accurate the human annotators were, we computed the precision considering only those data instances corresponding to defectively manufactured parts to avoid distortions due to the class imbalance.

After training the DRAEM and MLP models, we found that the DRAEM model achieved an AUC ROC of 0,8624, while the MLP surpassed it with an AUC ROC of 0,9894.

Table 5: Labeling quality and time measurements and the number of defects cases where the users could not tell if there was a defect or considered the defect did not correspond to existing classes. Best results are bolded, second-best results are highlighted in italics.

Experiment	Labeling quality			Labeling time (s)			Unidentified defects	
	Precision	Recall	F1	Median	Mean	Std. Dev.	Other	Unable to tell
1	0,2857	0,3576	0,3090	3	4,27	5,85	36,00	34,33
2	<i>0,9166</i>	<i>0,6304</i>	<i>0,7444</i>	2	3,08	5,67	<i>9,75</i>	21,00
3	0,6774	0,4983	0,5660	2	2,67	<i>5,38</i>	16,67	11,50
4	0,7339	0,4691	0,5693	2	3,00	5,71	10,67	8,00
5	0,8238	0,5212	0,6374	2	<i>2,83</i>	5,17	13,00	<i>7,75</i>
6	0,9393	0,8360	0,8831	2	3,22	5,82	1,50	5,25

To understand the effect of class imbalance and defect hinting on the manual revision process, we conducted six experiments, with 600 images each: (1) natural data imbalance; no defect hinting; (2) balanced dataset, complementing images of defective products with GAN-generated images; no defect hinting; (3) natural data imbalance; DRAEM anomaly map for defect hinting; (4) balanced dataset (same as Experiment 2); DRAEM anomaly map for defect hinting; (5) balanced dataset (same as Experiment 2); GradCAM heatmap for defect hinting; (6) balanced dataset (same as Experiment 2); defect hinting: nearest labelled image (considering the structural similarity index measure) and its label.

To evaluate how accurate the human annotators were, we computed the precision considering only those data instances corresponding to defectively manufactured parts to avoid distortions due to the class imbalance.

We asked the participants to execute one experiment per day to avoid labelling fatigue affecting their performance and influencing the results. We collected data regarding each experiment's start and end time, the time required to label each image, the assigned labels while having data on the ground truth labels, and whether the image was obtained from the dataset or synthetically generated. We found the best performance was achieved in Experiment 6 (see Table 5), which yielded good labelling times (a median of two seconds per image) while achieving the greatest precision, recall, and F1 scores.

Finally, we analysed whether the annotators' labelling precision decayed over time, by measuring the F1 score for every hundred images. We found it did, regardless of the experiment setting, with F1 scores falling at least 0,19 points from the initial score, within an hour of labelling. This finding confirms the relevance of data streams where images from the

printing pad are intertwined with synthetic ones to enable operators' attention monitoring and ensure the quality level of image annotation remains within acceptable thresholds.

4.2.5 Conclusions

This prototype presents a visual inspection pipeline that leverages machine learning models and active learning to partially automate the visual inspection task while collaborating with the user to determine the quality of the manufactured pieces the machine learning model is most uncertain about. We found that the best defect hinting strategy was to show the nearest known labelled image, which the annotator used as a reference. This resulted in a median labelling speedup of 33%, and increased precision, recall, and the F1 score by more than 0,5 points compared to the ordinary stream of data without defect hints. In future work, we plan to (i) compare labelling performance taking into account operators, artificial intelligence experts, and random people, to understand better whether labelling performance correlates with domain knowledge; (ii) explore new hinting techniques to improve the labelling process, and (iii) develop strategies and models for labelling consensus. We envision advanced models can be developed, considering users' expertise, performance, and current attention level to the task at hand.

5 NLP Functionalities

The term NLP comprises a broad spectrum of technologies and research topics. In the framework of this document, NLP covers topics such as knowledge graphs or active learning. Traditionally, NLP has also studied aspects that can provide the systems with functionalities that simplify the shop-floor operators' experience with AI systems, such as those that manage human-machine and machine-human operations.

This section attempts to provide a brief overview of the exercises performed, related to human-machine-human interaction, such as speech synthesis and recognition approaches, and then show how we have performed several proofs of concept to demonstrate that this functionality can be included in the architecture of STAR, complementing the interaction with other components. Finally, we mention the following steps, also within the NLP area, mainly related to the study of polarity and sentiment analysis of input messages.

5.1 Speech Synthesis and Recognition Approaches

Within the capabilities of speech synthesis and recognition approaches and focusing as much as possible on the functionality and the possibility of offering a fast and adaptable solution, we have focused on Speech-To-Text (STT) and Text-to-Speech (TTS) technologies.

Currently, the possibilities for implementing both are endless, in the form of services, libraries and components. In fact, one of the fundamental decisions is where to develop the STT/TTS, whether in the client itself with which the user interacts, in an in-house server or directly contracting a cloud service.

All three options, or even combinations of several, are valid and interesting: Using the browser or client for the STT and TTS is possible in modern browsers, it is sufficiently reliable and does not incur additional costs, doing it on an own server allows flexibility and third-party services provide excellent results.

Below we present some tests performed on the one hand using the Web Speech API in the operator's browser and on the other hand by using services on own and third party cloud servers.

5.1.1 Web Speech API for speech synthesis and recognition.

As mentioned, using the functions offered by browsers compatible with the Web Speech API¹⁰ for speech synthesis and recognition has considerable advantages in terms of less complexity of the necessary code and in terms of costs.

In the case of Speech Synthesis, the support is included by default in most modern browsers. Speech Synthesis is supported on Google Chrome, Microsoft Edge, Mozilla Firefox (only voices installed on the client machine), Opera and Safari for PCs and on Chrome, Firefox and Safari for Android.

¹⁰ https://developer.mozilla.org/en-US/docs/Web/API/Web_Speech_API

One of the most interesting points is that of the voice selection. The voices available for the synthesis depend on the used browsers and operating systems, but most spoken languages are supported (there are about 70 voices on Microsoft Edge and 30 on Google Chrome).

Microsoft Windows has a built-in speech synthesis engine with the ability to install voice packages from a list that includes over forty languages. The voices installed with this procedure are available in the list of voices offered by the Web Speech API.

The other items in the voice list refer to the browser manufacturer's online text-to-speech engines and require an internet connection. These voices, synthesized with techniques that make use of neural networks, are smoother, more pleasing and more human than those installed.

Regarding speech recognition, the support also comes by default in most modern browsers, and is a matter of implementing some support in Javascript and fine tuning the grammars to enable the functionality.

There are over sixty languages supported for recognition and different accents related to countries are available for the main ones. To help the system in the recognition process it is possible to define a grammar in JSGF format¹¹. Using grammar does not bind the recognition system.

The grammar structure requires a header in the format:

```
#JSGF version local char-encoding;
```

(where char-encoding and locale are non-mandatory), a grammar name in the format:

```
grammar grammarName;
```

and the body of the grammar where the expansion rules are contained:

```
public <ruleName> = ruleExpansion;
```

In the following table an excerpt of the JSGF grammar used in the proof of concept is shown:

Table 6: An example of JSGF grammar

```
#JSGF V1.0;
grammar utterances;
public <num> = one | two | three | 1 | 2 | 3 ;
public <value> = <num> thousand | 1000 | 2000 | 3000 ;
public <utterance> = Change demand <num> to <value> ;
```

The character " | " is used to define a set of alternative expansions. If a grammar is provided a comparison procedure is applied when applying the Speech to Text. The

¹¹ <https://www.w3.org/TR/jsgf>

comparison can be based on the Levenshtein distance (or similar metric) between the string extracted from the speech recognition and the elements present in the list of utterances, extracted from the JSFG grammar through a specifically developed function.

Among the sentences obtainable from the JSFG grammar in the previous example "Change demand 1 to 1000" is the one with the shortest distance from the example sentence recognized by the speech-to-text "change the man 12 1000".

Table 7: Example of calculating the distance between two strings

Step	String	Operation
1	change the man 12 1000	change
2	Change the man 12 1000	change
3	Change d h e man 12 1000	delete
4	Change de m an 12 1000	delete
5	Change deman 12 1000	add
6	Change demand 12 1000	add
7	Change demand 1 2 1000	change
8	Change demand 1 t 1000	add
9	Change demand 1 to 1000	
the distance between the two strings is 8		

The comparison function computes a threshold value beyond which it will not return any match with the grammar and the main program will return the raw text extracted from the speech-to-text.

With this definition of grammar and browser capabilities, we can improve the recognition of specific commands and limit potential recognition problems.

5.1.2 Cloud Speech synthesis and recognition

There are dozens of TTS and STT services in the cloud, in addition to the possibility of implementing your own using different libraries for speech processing and recognition.

In our case we have tested 3 of the main names, which we detail below. In our case and for the use cases of the project the results are similar, and it is only a matter of preference to choose one or the other. See below some comparisons.

Table 8: Text-To-Speech (Cloud)

Text-To-Speech (cloud)					
Google		Microsoft		AWS	
https://cloud.google.com/text-to-speech/docs		https://docs.microsoft.com/en-us/azure/cognitive-services/speech-service/		https://docs.aws.amazon.com/polly/latest/dg	
API – Client Libraries (C#, Go, Java, Node.js, PHP, Python, Ruby) – SSML		API – SDK (C#, C++, Go, Java, Node.js, JavaScript in a browser, Objective-C/Swift, Python)		SSML, AWS CLI, API	
Standard voice (from 0 to 4 million characters)	Wavenet voice (from 0 to 1 million characters)	Standard	Neural	Standard	Neural
\$ 4.00 / 1 million characters	\$ 16.00 / 1 million characters	\$ 4.00 / 1 million characters	\$ 16.00 / 1 million characters	\$ 4.00 / 1 million characters	\$ 16.00 / 1 million characters

Table 9: Speech to Text (Cloud)

Speech to Text (cloud)					
Google		Microsoft		AWS	
https://cloud.google.com/speech-to-text/docs		https://docs.microsoft.com/en-us/azure/cognitive-services/speech-service/		https://docs.aws.amazon.com/transcribe/latest/dg/	
API – Client Libraries (Python, Java, Node.js, C#, Go, PHP, Ruby)		API – Client (Windows, Linux, Docker) – SDK (C#, C++, Go, Java, Node.js, JavaScript in a browser, Objective-C/Swift, Python)		SSML, AWS CLI, API	
without Data Logging - default	with Data Logging opt-in	Standard	Custom	Standard	Call Analytics
up to 1 Million Mins (over: call for price)		\$1 per audio hour	\$1.40 per audio hour Endpoint hosting: \$0.0538 per model per hour	First 250,000 mins: \$1.44 / hour	First 250,000 mins: \$1.80 / hour
\$0.006 / 15 seconds **	\$0.004 / 15 seconds **			Next 750,000 mins: \$0.90 / hour	Next 750,000 mins: \$1.116 / hour
\$1.44 / hour	\$0.96 / hour			Next 4,000,000 mins: \$0.612 / hour	Next 4,000,000 mins: \$0.828 / hour
				Over 5,000,000 mins: \$0.468 / hour	Over 5,000,000 mins: \$0.684 / hour
** Each request is rounded up to the nearest increment of 15 seconds.				** Each request is rounded up to the nearest increment of 15 seconds.	
Volumes = minutes/month. Google (60 mins) and Microsoft (300 mins) services are free for small volumes)					

5.1.3 Architecture

In our case, a mixed architecture has been designed to cover the different options, although depending on the use case, one route or another can be chosen.

As can be seen in the architecture below, the web application (web-app) has a vocal component, this component is responsible for selecting the STT/TTS provider depending on the needs and preferences of the use case.

The main route, if the application is browser-based, and if the browser used by the PC or machine with which the operator is interacting supports it, is through the Web Speech API. In some cases, it is possible to use the HTML5 Audio module to support incompatible browsers and clients.

The alternative route is for specific cases where more control over the audio is required (for example because extra processing is required) or cases where the client application is integrated with a third party service. In the first case, TTS/STT can be implemented in a server using Mozilla DeepSpeech and similar libraries, in the second case the developer could make use of the APIs of any cloud service. It is important to mention that although sentiment analysis (SA) can be performed in the browser itself, the results are not particularly good, and it is common to implement SA on a (cloud) server where previously trained sentiment analysis AI models can be executed.

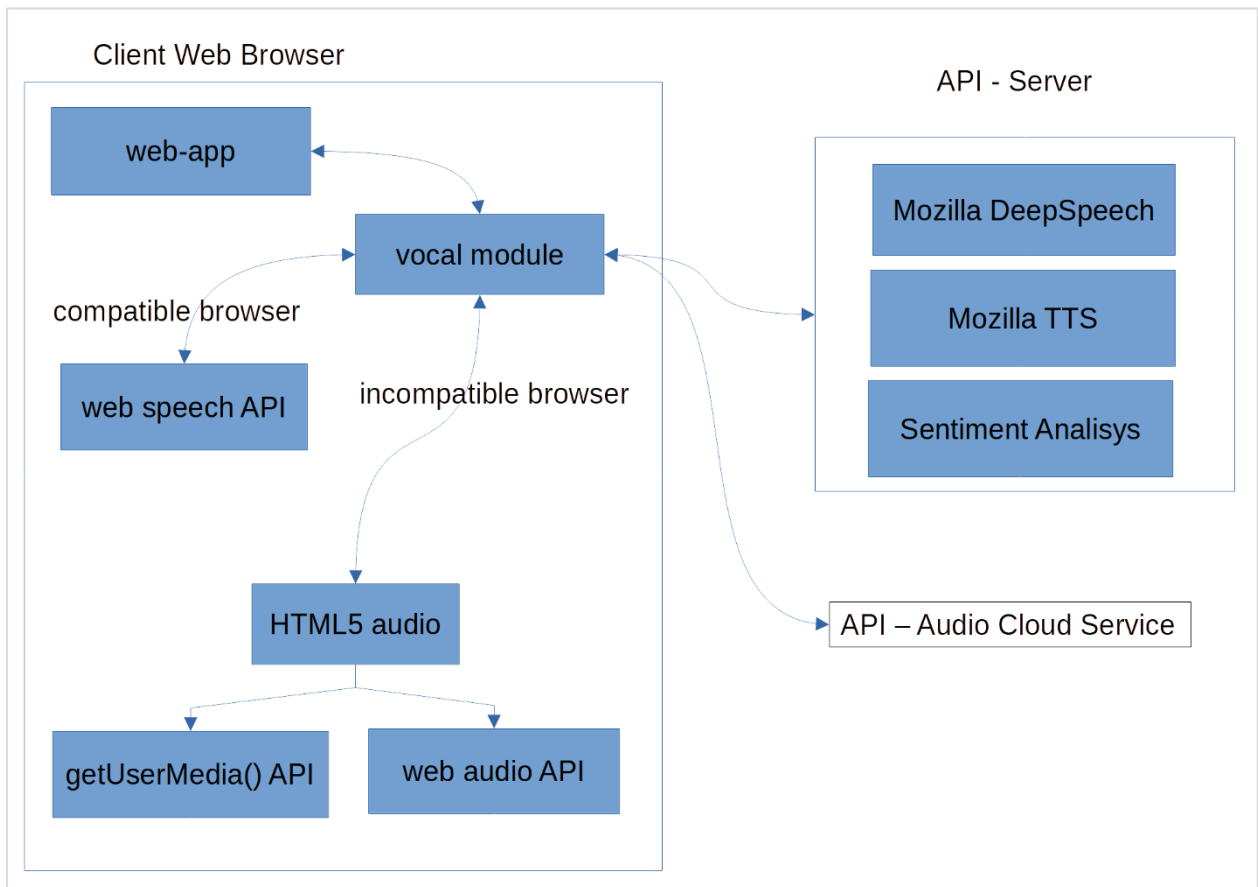


Figure 10: NLP Component Architecture

5.1.4 Proof of concept

In addition to the integration of STT and TTS for multimodal interaction in the demonstration that was presented in the first project review, we also performed proof-of-concepts of the different elements of the architecture shown above.

The image below shows an example of the proof-of-concept interface where different voices could be evaluated in different browsers and the recognition of various demand forecast related sentences could be tested. This implementation included versions with and without grammars, the former being the one that offered the best results by limiting the structures that could be recognized.

Web Speech API Demo 2

Speech Synthesizer:

Input text

Voice Microsoft Aria Online (Natural) - English (United States) (en-US) ▾ Synthesize

Recognition language English ▾

Utterances

- 1) Change demand [NUM] to [VALUE]
- 2) Explain demand [NUM]
- 3) Mark feature [NUM]
- 4) Start plan for demand [NUM]
- 5) Select option [NUM]
- 6) Show other options
- 7) Select feedback [NUM]
- 8) Utterance: Feedback [TEXT]

with [NUM]: [1, 2, 3], [VALUE]: [1000, 2000, 3000] and [TEXT]: *

Voice interaction

Activation words: "Ehi computer" | "Computer"

Start listening

Recognition results:

The user said:

Most likely utterance:

Figure 11: Web Speech API Demo

5.1.5 Next steps and sentiment analysis

As for the next steps, besides offering the possibility of including multimodal interaction in some of the pilots, we consider the study and implementation of sentiment analysis interesting. There are not many use cases that can benefit from it, but there are significant cases related to the detection of stress in workers or human-machine collaboration that may be of interest.

It is important to mention that these STT/STT technologies, or even those that depend on them such as polarity detection and sentiment analysis, are affected by the environment. In noisy conditions, these solutions may not be optimal, and shop-floors often are noisy. For this reason, there will be use cases that require the use of hands free microphones or one ear headsets, or in the context of environments where this is not possible for safety reasons, microphones or microphone arrays with noise reduction will be necessary.

However, it is also important to mention that an interaction system for an industrial environment has to support multimodal interaction, as in our case, allowing the user to extend or complement the information that could not be provided by voice.

In our case, this multimodality comes through voice support together with keyboard/mouse interaction and visual output, in cases where the operator requires it for comfort or noise problems.

6 Conclusions

This document has presented the work accomplished in STAR Task 4.3 and described 2 demos that integrate components from the whole WP4. Extensive work is described in even more detail in the publications related to the task (see Table 1).

Although early in the project, one of the prototypes is already strongly connected to a STAR use case and is already tested by the end-users, which is a nice achievement by itself.

In this deliverable, we presented the prototype architecture designed to acquire and encapsulate complex knowledge using semantic technologies and artificial intelligence. The system was instantiated for the demand forecasting use case in the manufacturing domain, using real world data. In particular, the system provides forecasts and explanations, enriches users' domain knowledge through a set of media news, recommends decision-making options, and collects users' feedback.

Furthermore, the system uses active learning to reduce manual labelling effort and better discriminate between good and bad media news reporting events related to the demand forecast domain. A series of experiments were executed to understand the best exploration and exploitation trade-off between strategies, which is required to learn from unlabeled media news entries while providing good recommendations to the users.

Additionally, we developed a visual inspection pipeline that leverages machine learning models and active learning to partially automate the visual inspection task while collaborating with the user to determine the quality of the manufactured pieces the machine learning model is most uncertain about. We found that the best defect hinting strategy was to show the nearest known labelled image, which the annotator used as a reference.

The next steps will involve:

- further research and experiments of the benefits of the active learning approach,
- exploitation of the developed techniques in other STAR use cases,
- assessment of feasibility to use the developed technology in other external use cases and integration into existing tools (e. g. QlectorLEAP),
- improvement of the NLP techniques in the industrial environment,
- connection of the developed AL approaches to the knowledge base population and with it optimization of the operators time and workload.

References

- [1] Kumar, P., & Gupta, A. (2020). Active learning query strategies for classification, regression, and clustering: a survey. *Journal of Computer Science and Technology*, 35(4), 913-945.
- [2] Budd, S., Robinson, E. C., & Kainz, B. (2021). A survey on active learning and human-in-the-loop deep learning for medical image analysis. *Medical Image Analysis*, 71, 102062.
- [3] Schröder, C., & Niekler, A. (2020). A survey of active learning for text classification using deep neural networks. *arXiv preprint arXiv:2008.07267*.
- [4] Samsonov, V., Lipp, J., Noodt, P., Solvay, A. F., & Meisen, T. (2019, December). More machine learning for less: Comparing data generation strategies in mechanical engineering and manufacturing. In *2019 IEEE Symposium Series on Computational Intelligence (SSCI)* (pp. 799-807). IEEE.
- [5] Meng, L., McWilliams, B., Jarosinski, W., Park, H. Y., Jung, Y. G., Lee, J., & Zhang, J. (2020). Machine learning in additive manufacturing: a review. *Jom*, 72(6), 2363-2377.
- [6] van Garderen, K. (2018). Active Learning for Overlay Prediction in Semi-conductor Manufacturing.
- [7] Dai, W., Mujeeb, A., Erdt, M., & Sourin, A. (2018, October). Towards automatic optical inspection of soldering defects. In *2018 International Conference on Cyberworlds (CW)* (pp. 375-382). IEEE.
- [8] Yue, X., Wen, Y., Hunt, J. H., & Shi, J. (2020). Active learning for Gaussian process considering uncertainties with application to shape control of composite fuselage. *IEEE Transactions on Automation Science and Engineering*, 18(1), 36-46.
- [9] Taylor, A. T., Berrueta, T. A., & Murphey, T. D. (2021). Active learning in robotics: A review of control principles. *Mechatronics*, 77, 102576.
- [10] Leban, G.; Fortuna, B.; Brank, J.; Grobelnik, M. Event registry: Learning about world events from news. In *Proceedings of the 23rd International Conference on World Wide Web*, 2014; pp. 107–110.
- [11] Ribeiro, M.T.; Singh, S.; Guestrin, C. "Why should i trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016; pp. 1135–1144.
- [12] Mollas, I.; Bassiliades, N.; Vlahavas, I.; Tsoumakas, G. LionForests: Local interpretation of random forests. *arXiv 2019*, arXiv:1911.08780.
- [13] Sundararajan, M.; Najmi, A. The many Shapley values for model explanation. In *Proceedings of the International Conference on Machine Learning*, PMLR, 2020; pp. 9269–9278.
- [14] International Conference on Web Search and Data Mining, 2021; pp. 373–381.

21. Yang, S.C.; Rank, C.; Whritner, J.A.; Nasraoui, O.; Shafto, P. Unifying Recommendation and Active Learning for Information Filtering and Recommender Systems; 2020; doi:10.31234/osf.io/jqa83.

[15] Crammer, K.; Dekel, O.; Keshet, J.; Shalev-Shwartz, S.; Singer, Y. Online Passive-Aggressive Algorithms. *J. Mach. Learn. Res.* 2006, 7, 551–585.

[16] Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; et al. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, 2020*; Association for Computational Linguistics: 2020; pp. 38–45.

[17] Lu, J.; MacNamee, B. Investigating the Effectiveness of Representations Based on Pretrained Transformer-based Language Models in Active Learning for Labelling Text Datasets. *arXiv 2020*, arXiv:2004.13138.

[18] Cer, D.; Yang, Y.; Kong, S.Y.; Hua, N.; Limtiaco, N.; John, R.S.; Constant, N.; Guajardo-Cespedes, M.; Yuan, S.; Tar, C.; et al. Universal Sentence Encoder. *arXiv 2018*, arXiv:1803.11175.

[19] Wilcoxon, F. Individual comparisons by ranking methods. In *Breakthroughs in Statistics*; Springer: Berlin/Heidelberg, Germany, 1992; pp. 196–202.

[20] : Zajec, P.; Rožanec, J.M.; Trajkova, E.; Novalija, I.; Kenda, K.; Fortuna, B.; Mladenić, D. Help Me Learn! Architecture and Strategies to Combine Recommendations and Active Learning in Manufacturing. *Information* 2021, 12, 473.

[21] Zavrtnik, V., Kristan, M., and Skočaj, D. (2021). Draem- a discriminatively trained reconstruction embedding for surface anomaly detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 8330–8339.

[22] Rožanec, J.M., Trajkova, E., Dam, P., Fortuna, B., & Mladenić, D. (2021). Streaming Machine Learning and Online Active Learning for Automated Visual Inspection. *ArXiv*, abs/2110.09396.

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

[24] Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D. (2017). Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, 618–626.

[25] Sampath, V., Maurtua, I., Aguilar Mart ´ın, J.J., and Gutierrez, A. (2021). A survey on generative adversarial networks for imbalance problems in computer vision tasks. *Journal of Big Data*, 8(1), 27. doi:10.1186/s40537-021-00414-0. URL <https://doi.org/10.1186/s40537-021-00414-0>.

[26] Liu, B., Zhu, Y., Song, K., and Elgammal, A. (2020). Towards faster and stabilized gan training for high-fidelity few-shot image synthesis. In International Conference on Learning Representations.