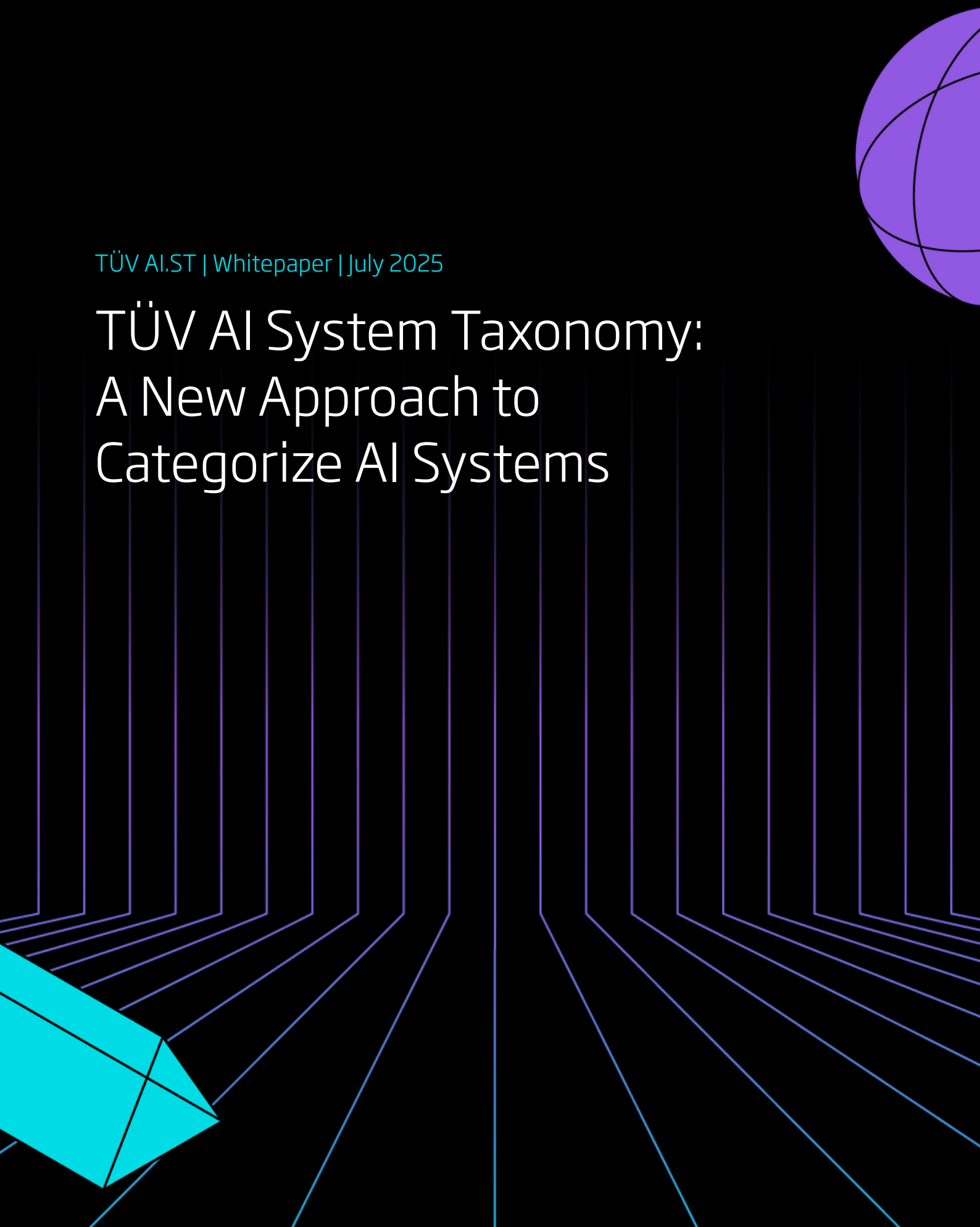
TÜV AI.LAB

TÜV AI.ST | Whitepaper | July 2025

TÜV AI System Taxonomy: A New Approach to Categorize AI Systems



Executive Summary

Artificial Intelligence (AI) is a term that encompasses a remarkably diverse range of technologies, systems, and applications. This broad scope creates a fundamental challenge: what exactly do we mean when we speak of 'AI'? Without a **shared understanding and clear terminology**, meaningful dialogue on AI, **expert AI assessment**, and compliance with AI regulation become increasingly difficult. The TÜV AI System Taxonomy addresses this problem by introducing **a new, structured taxonomy for AI systems** – one that brings conceptual clarity to a complex and fast-evolving field. The TÜV AI System Taxonomy thus enables everyone categorizing AI systems to speak the same language.

The goal of the TÜV AI System Taxonomy (version 0.1), is to enable a common understanding of AI applications across the AI community. It is designed to **group similar applications, clearly define essential terms, and create a functional vocabulary that supports dialogue, innovation, and oversight**. This taxonomy achieves more than just standardizing terminology – thereby creating a common language which is essential to be sure that we, in fact, mean the same things by our words. In addition, it creates the **potential for new insights by revealing structural patterns** across AI systems, supporting both academic inquiry and practical application. For companies that develop or use AI, this taxonomy can be used to **structure their own repository of AI systems**. Moreover, it has also been developed in the context of the **TÜV AI Assessment Framework**. This framework, among other things, aims to match AI applications with the testing and certification tools most appropriate to their functional characteristics. To achieve this, the taxonomy must do more than categorize – it must guide action.

The TÜV AI System Taxonomy (AI.ST) is built on **three categories: task, input, and implementation**. These were not chosen at random, but carefully selected based on rigorous criteria for what constitutes a useful and sustainable classification system. At its core lies the **principle of the functional unit**: What functionality does an AI system implement? Based on what kind of input? Through which underlying method or algorithm?

Each category plays a distinct and essential role. A careful selection of basic tasks represent fundamental units of AI functionality, and can be combined to describe even complex systems – offering a **scalable structure that can evolve alongside the field**. Inputs, then, are critical because they form the operational basis for all AI functionality. The implementation, finally, is what defines an AI system as such; it distinguishes AI from classic rule-based software through its underlying algorithmic approach.

Furthermore, the paper addresses key conceptual challenges, such as how to delineate the boundaries of an AI system, and how this relates to definitions within regulatory frameworks like the **EU AI Act**.

Ultimately, we believe this taxonomy provides real value to anyone seeking to engage with AI in a precise and structured manner. For TÜV AI.Lab, the AI.ST serves as a central asset in our mission to enable **rigorous, reliable, and scalable assessment of AI systems** – contributing to a **safe and trustworthy future for AI**.

The Need for a Shared Language in the AI Debate

The term Artificial Intelligence (AI) has become an ubiquitous fixture in public discourse, policy debates, corporate strategy, and academic inquiry in recent years. However, using just the term 'Artificial Intelligence' implies a kind of unity that can obfuscate the intrinsic differences of the field. The term can be used to describe such different applications as, for example, autonomous vehicles, generative language models, medical devices that detect cancer in MRI images or the applications that turn a scanned, handwritten document into machine-readable text. Beneath the umbrella term 'Artificial Intelligence' lies a vast and heterogeneous landscape of technologies, methodologies, and applications that vary significantly in their underlying principles, intended purposes, and risk profiles.

The increasing reliance on AI in critical decision-making processes has also brought with it the urgent need for systematic approaches to evaluation, regulation, and certification. With the final publication of the EU AI Act in August 2024, the European Union has set out to establish the first comprehensive legislation to regulate the development and use of safe and trustworthy AI. Yet, meaningful oversight is difficult when the term 'AI' is used as a catch-all phrase, masking the nuanced differences between, for example, a simple rule-based expert system and a deep learning model trained on vast datasets. Such generalizations hinder clarity, impede the development of appropriate standards, and create ambiguity in discussions about central concerns around AI, such as, for example, ethics, safety, and accountability.

To move beyond superficial categorizations, there is thus a pressing need for a new taxonomy that reflects the diversity within AI. This taxonomy must enable stakeholders, such as researchers, developers, regulators, auditors, and the public, to distinguish between types of AI systems, understand their specific characteristics and implications, and apply tailored procedures for testing, assessment, certification, and governance. Without

such a structure, discussions about AI are at risk of becoming increasingly fragmented and incoherent, with potential consequences for both innovation and societal trust.

In conjunction with the AI Assessment Matrix, this new taxonomy is also part of the AI Assessment Framework developed at the TÜV AI.Lab. Using the AI Assessment Matrix, auditors and testers can determine which aspects of the AI system need to be evaluated; the AI Assessment Matrix is a guide to the relevant test dimensions, areas and modes. The AI System Taxonomy then specifies how this aspect of the AI system can be tested, by collecting and ordering methodologies and metrics relative to the specific characteristics of the AI system. In that way, the AI System Taxonomy functions as a categorization framework, allowing testing and certification tools to be assigned to those AI systems for which they are applicable. In practice this could mean that for instance, the AI Assessment Matrix shows that an AI system must satisfy performance requirements during the verification and validation phase of the AI system life cycle. Depending on the task performed by the AI system, these requirements could be satisfied using either an accuracy metric (for AI systems performing a labelling task) or the mean absolute error (for regression tasks), amongst other possible metrics, which would then be stored under the relevant category of the AI System Taxonomy. In this way, this new taxonomy for AI can be a resource for both manufacturers and auditors for choosing the appropriate methodologies and processes to test a specific AI system. Moreover, we believe that the AI System Taxonomy is flexible and scalable in such a way that it will be compatible with any official standards developed in the future that regulate the testing and certification of different AI systems.

The State of the Art

There is currently little systematization in the AI landscape when it comes to a comprehensive and consistent classification of different AI systems. The most common distinction to be made is the one between supervised, unsupervised and reinforcement learning

in machine learning[1]. This distinction does capture some important intrinsic differences in different AI applications, but there is still a great deal of variance in the individual categories. For example, fraud detection, speech recognition and disease diagnosis can all fall under the umbrella of supervised learning. Moreover, this differentiation only applies to machine learning applications, which are a subset of Artificial Intelligence, but does not include logic-based systems or search and optimization methods.

Another common way to distinguish between different AI systems involves categorizing them based on their primary functional domain[2]. For example, natural language processing (NLP), computer vision and anomaly detection can be considered their own subfields within the larger domain of AI and much research is focused on these subfields. However, for the purpose of an exhaustive AI categorization, this approach also has its limitations, namely that each domain can subsume a myriad of heterogeneous lower-level tasks; for instance, natural language processing alone encompasses a range of tasks such as translation, summarization, and sentiment analysis, which each represent distinct functional processes, yet are collectively subsumed under a broad umbrella term. Moreover, a significant challenge with this approach also lies in finding an exhaustive list of such high-level domains under which all kinds of AI applications can be subsumed.

Further distinctions between different AI systems can be made based on their underlying implementation. This approach is prominent in the standardization literature, where different norms and standards address the performance and robustness of neural networks to just mention one example[3]. However, this strategy also has certain limitations, namely that the technical boundary between AI and non-AI applications is quite fuzzy. While some architectures, such as neural networks and expert system-based approaches, very

[1] See for instance: Jordan, Michael I., and Tom M. Mitchell. "Machine learning: Trends, perspectives, and prospects." *Science* 349.6245 (2015): 255-260.
[2] See for instance Pouyanfar, Samira, et al. "A survey on deep learning: Algorithms, techniques, and applications." *ACM computing surveys* (CSUR) 51.5 (2018): 1-36.
[3] See for instance the series of standards under ISO 24029 *Assessment of Robustness of Neural Networks*

clearly fall under the AI paradigm, it is less obvious for other techniques, such as basic regression and optimization algorithms. This is an important issue, especially in the light of the EU Comission's guidelines to the definition of AI systems in the EU AI Act, which do not explicitly exclude these kinds of algorithms.

Objectives: A Consistent Taxonomy for AI Systems

The objective of this whitepaper is to introduce greater coherence into the currently fragmented landscape of AI system classification and to develop a new taxonomy for AI systems that is useful across the board: it is integral to the TÜV AI.Lab AI Assessment Framework and can be used by auditors and testers, but the aim is also to have a broader impact on the AI landscape as a whole. In the end, everyone, from developers and manufactures, to auditors and tester, to even the general public, can profit from a system that gives a systematization and a language to talk about different AI systems in well-defined distinct terms. As AI technologies continue to permeate diverse aspects of daily life, the imperative for clear distinctions and categorizations becomes ever more pressing – a problem for which the AI System Taxonomy provides a possible solution. Nevertheless, its core purpose as part of the TÜV AI.Lab AI Assessment Framework consists of ensuring that AI testing and certification tools can be mapped to the most suitable applications, thereby assisting developers and auditors in finding the right tools to establish the safety and trustworthiness of AI.

A taxonomy consists of a set of categories that in turn each contain a list of elements. For each category, one (or more) elements are selected to characterize an AI system, with the goal of ensuring that the characterization is as complete and unambiguous as possible. However, the requirements for such a new taxonomy of AI are manifold and not all of them can be fulfilled simultaneously. Some desiderata for a successful AI system taxonomy may include:

- **Unambiguity:** Each AI system maps to *one combination of elements.*
- **Exhaustiveness:** *Every possible AI system* can be characterized using the taxonomy.
- **Singularity:** Each AI system maps to exactly *one element per category.*
- **Injectivity:** Each combination of elements maps to *one unique system* .

Not all of these desiderata are of equal importance. A taxonomy must be as unambiguous as possible, otherwise it fails its purpose to classify objects into distinct categories. The purpose of this taxonomy is also to be exhaustive - as previously stated, a major problem with existing categorization attempts is that they cannot be applied to all AI systems. Consequently, the taxonomy was designed to be as unambiguous and exhaustive as possible.

However, when it comes to the other two desiderata, they can actually hinder the applicability of such a taxonomy to real-life AI systems. For example, a functional view of the task of the system implies that one system can have more than one task, plus some modern AI systems are far too complex to be narrowed down to just one element per category, which is why the AI System Taxonomy is not singular. This argument will be explored in detail in the subsequent chapters.

Moreover, the taxonomy is designed in such a way that different AI systems can be assigned the same elements in each category; it is therefore not injective. Considering the myriads of possible AI applications, finding an injective taxonomy can be considered a virtually impossible task. Therefore, the aim of this taxonomy is to find categories and elements such that those systems that fall under the same categorization are merely similar enough. This similarity applies to the functioning of the AI system and how this functioning is implemented through a specific algorithm. For example, a system that scans MRI images for brain cancer and one that scans X-Rays für bone fractures will most likely have the same assignments in this taxonomy, since on a functional level they do very similar things: labelling image data, most likely using a convolutional neural network to do so. In many cases AI systems that are similar in this way can also be tested in similar ways.

Rejecting injectivity therefore promotes the scalability of this approach when it comes to auditing and testing AI systems.

Lastly, another consideration that prohibits injectivity is the complexity of the resulting taxonomy. It would be possible to add more categories and more elements to further delineate different AI systems, but this is an exercise that will only result in a taxonomy that is as big as the space of possible AI applications itself. The challenge lies in finding a taxonomy that is as comprehensive as possible, but only as complex as necessary. In conclusion, the proposed new taxonomy of AI systems satisfies two of the four proposed criteria. It is unambiguous and exhaustive, but neither singular nor injective.

The AI System Taxonomy: Task – Input – Implementation

The new taxonomy for AI systems has three categories: task, input and implementation. In this section we will delve into the reasons why these three categories were chosen and how they each impact AI assessment and testing. However, before this, first a note on what kind of objects this taxonomy applies to. So far, we have used both the terms 'AI systems' and 'AI applications' interchangeably. We will continue to do so but feel the need to clarify the term 'AI system' in particular, as it is a loaded term because it is used in the EU AI Act and has a very specific definition there. When the term 'AI system' is used in this whitepaper, it is not necessarily meant in the same way as in the EU AI Act (though it could be), chiefly because some details of what is meant by an 'AI system' in the EU AI Act, particularly in relation to the system boundaries of an AI system, are at the time of this publication still unanswered. Consequently, when this whitepaper refers to an 'AI system' or 'AI application' as a target of this taxonomy, we mean a **functional unit** that is implemented with AI techniques[4]. This is because this taxonomy is focused on the functionalities that are implemented through AI and that are specific to AI.

Due to the uncertainty around the terminology of 'AI system' in the EU AI Act, this work cannot strictly adhere to the terminology laid out in the EU AI Act yet, but it is also nowhere in explicit contradiction and thus wholly compatible with it. Moreover, this new taxonomy for AI systems is also informed by other regulatory principles derived from the AI Act. One such principle is the primacy of intended purpose. This principle, which is very important for certification, captures what the product claims to do and to which degree it

[4] This functional unit, and thus target of this taxonomy, could also consist of an (AI) component of an AI system (where AI system is understood in the sense of the AI Act). However, since both the terms 'AI component' and 'AI system' have no precise accepted definition at this point in time, we will not further engage in the discussion of what constitutes an (AI) component (in relation to an AI system) and instead leave this question open for some other time.

actually does that. A similar principle should hold for assessing AI systems. As a consequence, in this new taxonomy of AI systems, a primary category is the **task** of the AI system. The task describes the function of the AI system – what it is intended to do – and is therefore central to any characterization of an AI system, especially one that places the functional unit at the centre of the taxonomy. Moreover, on a pragmatic level, many existing tools to test and assess AI systems are task-specific, such as different metrics depending on whether the system is primarily used for classification or regression.

The second category in the new taxonomy for AI is the **input** to the system. The input describes what is fed into the system, and can serve to further explicate its function. For example, an AI system whose task is classification might be further specified by differentiating between text and image classification, depending on whether the input is text or images (or more specifically an array of characters or of pixel values). Input is critical for any characterization of an AI system, as it constitutes the basis of any further functionalities, and therefore what goes into the system is a fundamental feature of any system. Moreover, how AI systems are evaluated can also change based on different input types; for instance, when assessing the explainability of a classification task, counterfactuals are more appropriate for tabular data input than for image data. Together, task and input give a comprehensive answer to the question what the function of the AI system is and what basis it operates on.

The third category of the new taxonomy for AI is **implementation**. While task and input describe *what* an AI system does and on what basis, the implementation describes *how* it does that. Implementation refers to the underlying models and algorithms that implement the functionality of the AI system, such as, for example, neural networks or random forest architectures. Moreover, a further reason why the implementation is such a central part of the characterization of an AI system is that it is only because of the underlying algorithm that we can speak about AI applications in the first place. AI has come about as a

series of methods and techniques that differ fundamentally from a standard deterministic software architecture. The implementation with AI-specific algorithms and architectures is the defining element that transforms a computational system into artificial intelligence. In the following, every category of the new taxonomy for AI will be presented in detail and its elements will be introduced.

Task

The task describes the functional relationship between the input and the output of the AI system. What is meant by the functional relationship is the steps and manipulations it takes to get from the input to the output. Therefore, despite not having an explicit output category in the present taxonomy, the output is subsumed under the task. This is particularly true for non-generative tasks: for example, the output of a labelling task is a label, the output of a clustering task is a set of clusters, the output of a regression task is a (multidimensional) value and so on. For generative tasks the situation is different as the task of generation does not imply any particular output. Therefore, part of the task description is the output format; any generation-transformation and generation-creation tasks need to be further specified by also including the output format. The options for the output format are the same as for the input as within these categories, there are no significant limitations to what generative AI can produce.

That is not to say that the output and its specific format does not matter at all when it comes to the testing of a specific AI system. For instance, assessing a neural network that performs a classification task could involve calculating metrics, such as accuracy, on the final output of the network, or it could also consist of evaluating the calibration of the network by looking at the probability vector that is the output of the last layer of the network. Alternatively, a particular method might only work on a certain file format but not on others for computational reasons. In response, one could argue that the specific output format of an AI system should be included after all. However, similar to the arguments

against injectivity above, further specifying the output format would firstly result in a decrease of generalizability of the current approach and secondly, present the challenge of coming up with a complete categorization of output formats. While the latter is simply not feasible in practice, the former would make the taxonomy objectively weaker conceptually. That many assessment and testing methods fall into the same categorization cannot be avoided due to the complexity of the AI landscape and the vastness of possible applications. Staying at the task level presents an acceptable reduction of this complexity, whereas including a further output dimension would introduce redundancies and make the taxonomy unusable.

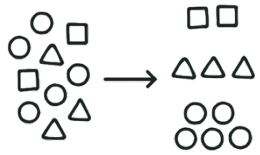
The following Table 1 provides a list of the different tasks that constitute the elements of the task category. Some tasks stand on their own, such as segmentation, identification, regression and so on. For others the task itself includes a further specification, e.g. binary and multi-class labelling, or generation-creation and generation-transformation with the relevant outputs, e.g. generation-creation-unstructured text. All outputs can be applied to both generation-transformation and generation-creation tasks. For the sake of brevity, the naming convention for the generation task is the following: generation-(creation/transformation)-(ultimate output)[5]. Ultimate output refers here to the most detailed level of the output category, e.g. 'unstructured text' instead of 'text-unstructured text'. A generation application that can produce multiple types of outputs can use the 'multimedia' designator as the respective output category.

[5] Some examples: generation-creation-unstructured text, generation-creation-graph data, generation-transformation-static 2D image, generation-transformation-structured text

Table 1: Elements of the Task Category

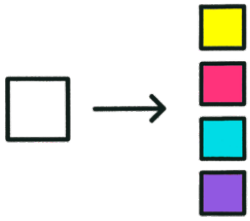
TASK	SPECIFICATION	OUTPUT	
Clustering	-	-	
Future State Prediction	-	-	
Identification	-	-	
Labelling	Binary	-	
	Multi-Class	-	
Ranking	-	-	
Regression	-	-	
Retrieval	-	-	
Segmentation	-	-	
Generation	Creation	Image Data	Static 2D Image
			Static 3D Image
		Text	Token
	Transformation		Structured Text
			Unstructured Text
		Audio Data	
		Video Data	
		Structured Data	
		Time-Series Data	
		Graph Data	
		Multimedia	

The following list provides a definition for each type of task, as well as an illustration of these definitions:



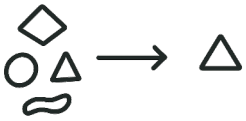
Clustering

The task is to group the input into clusters of relative similarity.



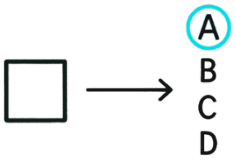
Future State Prediction

Based on the input, (one or multiple) possible future states of are calculated. In this case, the input usually refers to the current state of a (physical or non-physical) system that the AI system applies to.



Identification

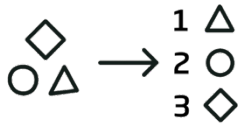
The task is to pick elements from the given input based on some relevance criteria.



Labelling

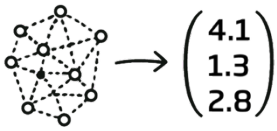
The task is to find a label for the input. 'Binary' and 'multi-class' refer to whether there are just two possible output labels (binary) or more (multi-class).[6]

[6] It is important to note that here and in the other tasks, 'the input' refers to the *whole* input that is available at this stage of the processing, where the whole input can refer to a single complete entity (e.g. for labelling, future state prediction, segmentation and generation-transformation) or to multiple entities at once (e.g. for ranking, clustering and identification). This is important since the target of the task can determine the task categorization. For example, in image classification, the goal is to determine whether the image *as a whole* contains an object or not, so the target is usually the whole image. In contrast, in a similar object detection application, the goal is identifying only *parts of the image*, which is why the image must first be segmented into its different parts and then one of these parts must be chosen before it can be labelled (For a more detailed explanation see the examples later on)



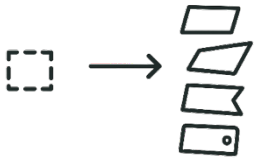
Ranking

The task is to arrange the input into an order according to some relevance criteria.



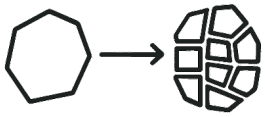
Regression

The task is to detect relationships between the input data and based on that calculate (multidimensional) numerical values.



Retrieval

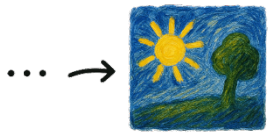
The task is to search for and return relevant entities based on the input where the search space consists of more than just the immediately provided input; the returned entities are not or only insignificantly altered. [7]



Segmentation

The task is to split the input into multiple different singular entities.

[7] One might wonder what exactly separates retrieval from identification, when the goal of both is, broadly speaking, to pick one or more entities based on the input. Essentially, the difference lies in whether the space from which the relevant entities are picked is part of the explicitly provided input (in which case, this task should be called Identification), or if the AI system has access to a wider range of sources, for example whole databases or the internet (e.g. through APIs) as part of the general system specification. In the second case, if the task is to pick entities from these external sources and not from the explicitly provided input, this is called retrieval. In this case, the input then usually consists of the criteria according to which the relevant entity is picked. This is for instance the case for AI-based recommendation algorithms or RAG applications.



Generation – Creation

The task is to create output based on the input where the output is new/original to a significant degree; generation-creation is further specified with different output formats

Generation – Transformation

The task is to create output based on the input where the output is a transformation of the input (can refer to both content and/or form); generation-transformation is further specified with different output formats

A range of examples of AI systems that implement one task are given in Table 2. These are not complete descriptions of an AI system but rather the breakdown of more high-level tasks into the more fundamental tasks of this taxonomy. It is immediately evident that the conception of task as introduced here is able to subsume applications under a single task that might previously have been understood as different tasks (e.g. image classification and sentiment analysis) due to their different domains. This break-down of higher-level tasks into more fundamental tasks entails a necessary reduction in complexity that enables systematic assessment of analogous applications.

Moreover, two things can be immediately noticed with this conception of task. Firstly, there is no 'classification' task even though this a common category of AI application. Instead, what is usually referred to as 'classification' is here subsumed under the 'labelling' task. That is because 'labelling' refers to any task that produces a 'label' as an output. This label can be the selected class of a classic classification task, but it also extends beyond

Table 2: Single Task Examples

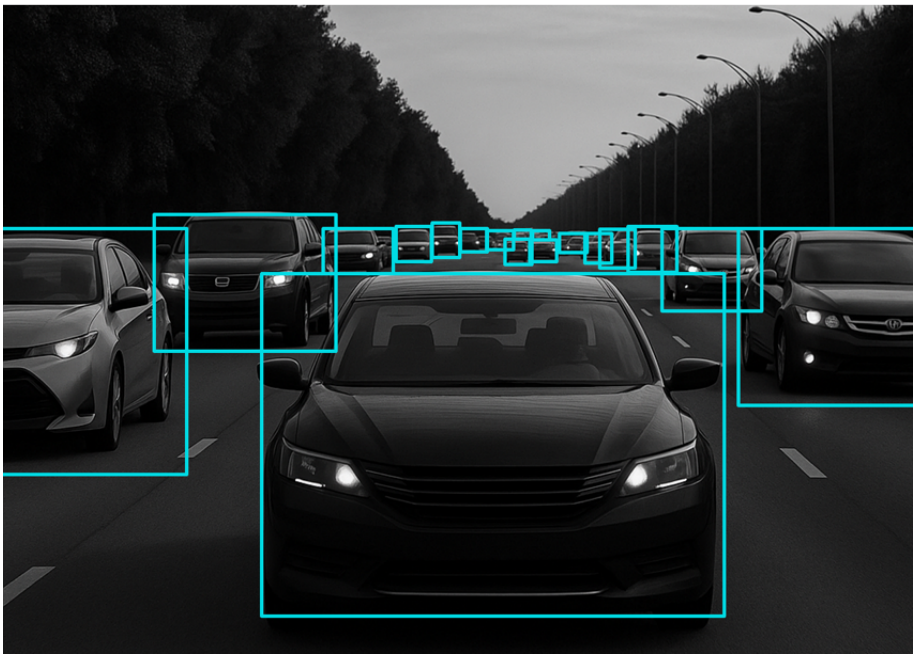
EXAMPLE APPLICATION	TASK
Image Classification	Labelling
Sentiment Analysis	Labelling
Language Identification	Labelling
Emotion Recognition	Labelling
Sentence Boundary Detection	Segmentation
Tokenization	Segmentation
Image Segmentation	Segmentation
Prompt Answering	Generation-Creation
Image Generation	Generation-Creation
Translation	Generation-Transformation
Transcription	Generation-Transformation
Image Reconstruction	Generation-Transformation
Sound Denoising	Generation-Transformation
Text Summarization	Generation-Transformation
Semantic Clustering	Clustering
Time Series Forecasting	Regression
Trajectory Prediction	Future State Prediction

that to include other AI tasks that produce a label but are not commonly referred to as classification. An example of that would be sentiment analysis, which is usually grouped under natural language processing, but whose end goal is to produce a tone indication for a given text input in the form one of a label: 'positive', 'negative' or 'neutral' (with more options available for more refined sentiment analysis tasks).

Secondly, there are some tasks on this list that in general do not appear on many AI task lists, for example 'Future State Prediction' or 'Retrieval'. The reason for this lies in how the assignments of task elements to AI systems is actually supposed to work. A general problem for any classification of AI tasks is the sheer volume and complexity of possible tasks that

AI applications can carry out. The technology is getting more refined and as a consequence, the field of possible applications becomes broader and broader. Finding a way to bring order into this field where each possible AI application, from simple classification to natural language processing and complex systems used in autonomous driving, can be subsumed under just one task can therefore be considered almost impossible. Instead, this classification views certain applications as composite objects, while leaving open the possibility that some AI system can indeed only have one task. This ability to combine multiple more basic tasks to a complex composite task is a huge advantage of this taxonomy, as it allows it to be flexible with regards to future developments in AI and possible new applications. The possibility to combine more basic tasks into complex ones allows this taxonomy to cover a wide variety of conceivable AI application and we are confident that even if AI develops more advanced capabilities in the future, these can also be subsumed under this framework.

For the composite tasks it is important to consider how the input is manipulated in different stages to arrive at the output on a conceptual functional level. What we mean by that is best illustrated with some examples: take for instance object recognition with image input where the final goal is to identify the position of all cars in the image. This object recognition is conceptually not an immediate action on the whole of the input image; the identification involves firstly the segmentation



AI-Generated Image: Vehicle recognition in road traffic.

of the input into its different parts and secondly, the identification of those parts that are relevant to the task. Both steps are essential as the goal is to both pick out only one kind of object (in contrast to recognizing and naming all objects in the image) and to pick out the correct objects (the cars and not, for example, the bikes). Therefore, in order to be able to identify all cars in the image, several steps have to be taken beforehand: one must first segment the image into its different parts (Segmentation) and then one must identify the relevant ones, i.e. the car-like objects (Identification). In total, this object recognition is then a combination of two tasks: segmentation and identification.

In contrast, a simple multi-class image classification task where the goal is merely to tell whether an image contains one or multiple cars, but not determine the positions, can be classified as a pure binary labelling task. This is due to the fact that the whole image is the subject of classification; there are only two options: either the label for the whole image is „contains car" or „does not contain car", no further refinement on the input and the result of the task is possible. This example also illustrates the difference between 'labelling' and 'identification' tasks: a labelling task produces a (natural language) label as the output – in this case "contains car" or "does not contain car". In contrast, the output of the identification task is not a label, but a specific part of the input. Therefore, these are different tasks, despite the fact that the criteria for picking out a label, versus picking out that part of the input, can be thought of as being the same: a car being in the image (though whether the AI system has an internal representation of the 'car' concept is up for debate). The main point is that what is returned as output is conceptually different in labelling and identification tasks.

Another example for composite tasks of AI applications is gameplay. AI systems like DeepMind's AI chess applications or AlphaGo evaluate the current state of the system and pick an action based on which future state is deemed most likely to win. Therefore, these tasks can be subsumed under future state prediction + identification, since future states

are calculated based on the current state. Then one of the possible future states is identified based on its likelihood to lead to a win (or some other relevance criteria).

However, the previous examples are just very general examples and the correct task assignments will depend on the exact specification of the general system specifications of the AI application in question. For example, an application that detects offensive speech in a text can differ in its task assignments – depending on whether the intended output is merely a yes/no label, indicating whether or not the input text contains offensive speech, or if the application should also return the text segments it has identified as offensive speech to the user. In the first case, the application is counted as a pure labelling task because it performs the task (assigning a label) on the whole of the input. However, in the second case several steps are at play, similarly to object recognition: in order to retrieve the offensive speech in question, the whole text must first be divided into segments (Segmentation), from which an individual segment is chosen according to some offensive speech relevance criteria (Identification) returned to the user.

The task can also depend on the specific variation of the input. By that we do not mean the general type of input, which is specified in the second category 'input' later on, but how, within one type, different variations of input and how they relate to the functionality of the AI system as a whole can influence the task classification. For example, speaker gender identification is an application that can have different task assignments, depending on whether the input is an audio recording of just one person or if there are multiple speakers in the audio and the application is supposed to determine the gender of every speaker in the audio. Both cases would have 'Audio Data' as the input type[8] but in the first case the whole input is the target of the task, which is why it can be classified as a labelling task. In the second case, before the gender of the speakers can be assigned, the input must first be segmented into the different parts corresponding to each speaker. Then a further distinction must be made as to whether the application should also be able to

[8] See next chapter for an explanation

identify when the same speaker spoke multiple times – and thus subsume all utterance of that speaker under one gender assignment – or if the application should merely assign a gender to each segment regardless of whether the speaker in question has spoken before. In both cases the output is a label indicating the gender of the speakers and thus they end in a labelling task, but in the first case a further intermediate step must be taken, namely the mapping of individual segments to one speaker. This mapping can be seen as an identification task because it picks from the set of all segments only those that belong to the same speaker, with 'belonging to the same speaker' being the relevance criteria in this case. The output of both applications will probably look very similar: a segmentation of the audio input into different parts corresponding to a switch in speakers and a gender label for each segment. However, the functional intent behind these applications is different, which is why they have different task assignments. One goes the extra step to identify when the same speaker has spoken, whereas the other merely labels each segment, disregarding possible duplications. It is important to note that in both cases, the relevant audio parts are provided with a label (the gender of the speaker), which is why this is not merely an identification task as in the examples above, but a labelling task. In the end, depending on variations in the input and the exact task specification, this speaker gender identification application could be classified in three different ways for the task category: a) labelling, b) segmentation + labelling, or c) segmentation + identification + labelling.

Lastly, it is important to note that in this taxonomy, the technical implementation of the task is irrelevant for the task classification. For instance, sentiment analysis can be implemented based on the supervised learning paradigm, where text and corresponding sentiment label pairs are fed into a neural network. A different way to implement sentiment analysis involves clustering algorithms that cluster words according to semantic similarity and assigning sentiment labels to the clusters. The overarching sentiment of a text can

then be determined according to the relative size of the clusters. However, in both cases the task remains the same: assigning a sentiment label to the whole input. Therefore, both cases count as a labelling task.

Input

The input dimension of the taxonomy refers to the *initial input* to an AI system that enter the system through a dedicated interface; this could be a user interface where the user has to manually enter the input, or the interface to other systems if the AI system is embedded into a larger system of systems, for instance in autonomous driving, where the input might consist of input from sensors in the vehicle. It is important to note that the input dimension does not refer to any intermediate state between the initial input and the eventual output that might be generated as a result of processing the initial input as part of the functionality of the AI application and as captured, for example, as one part of a composite task.

This category aims to be as descriptive as possible with its elements to avoid ambiguity. There is some additional structure to some elements, i.e. images and text, but the elements that can ultimately be assigned to the AI system are the elements at the most detailed level of each row of the structure (represented in Table 3), e.g. 'Static 2D image', 'Unstructured Text', 'Token' or 'Structured Text'. Since most elements in this category are descriptive, they do not require much of a definition. 'Structured Data' can refer to tabular data or other types of data that has a predetermined structure, i.e. by originating from a sensor and being clearly marked as such as part of the metadata. This could be the case for AI systems that are integrated in autonomous vehicles. However, in such cases it is

Table 3: Elements of the Input Category

INPUT	SPECIFICATION
Image Data	Static 2D Image
	Static 3D Image
Text	Token
	Structured Text
	Unstructured Text
Audio Data	-
Video Data	-
Structured Data	-
Time-Series Data	-
Graph Data	-
Multimedia	-

highly likely that the timestamp is also an important part of that sensor data, in which case it becomes an instance of time-series data instead.

The text category is split into tokens, structured and unstructured text. Tokens refer to sub-word units of text and are often used in natural language processing. Tokens can be the input to an AI system if a tokenization of text has already been done as part of preprocessing activities that utilized classic non-AI-based NLP techniques. The tokens can then be fed into the AI system where they are further processed to achieve certain task-specific ends. The text category is further split into structured and unstructured text. Structured text refers to text that exhibits a consistent, (partially) predetermined format, for example JSON and Markdown files or code snippets. Different instances of the same kind of structured text are built using the same rules and regularities and therefore share commonalities in syntax. In contrast, unstructured text refers to text that does not possess this consistent, (partially) pre-determined format and every instance of unstructured text can be completely different from each other.

Moreover, there is a difference between multiple inputs and one input of the 'multimedia'-variety. As with the task category, the input category also allows for the assignment of multiple input elements to one AI system. For example, one can imagine an AI application that allows for both image input and a text prompt that specifies what the application should do with the image. Such a system would have both unstructred text and static 2D

image input, because while the inputs might be entered into the same interface, they are separate entities. In contrast, some inputs count as 'multimedia' inputs when just one input is entered which in itself contains multiple disparate entities and which is not structured in a way that could classify it as structured data. An example for this is a document that contains both text and images.

Implementation

The underlying architecture and algorithm of an AI system is essential for its designation as an AI system, which is why the third and final category of this new taxonomy for AI systems is their implementation. This category poses several challenges: the field of AI algorithms and architectures is extremely broad and ever evolving, the elements are highly diverse, and there is little obvious structure to organize them. It is also difficult to determine the right level of detail – too much granularity risks fragmentation, while too little reduces usefulness. Similar to the input category, the element that can be assigned to the AI-system is an element at the most detailed level (e.g. 'Feedforward Neural Network', not just 'Neural Network').

The central concept here is again the functional unity: different kinds of implementation that are based on similar functional principles are grouped together wherever possible. For instance, architectures that are built on the same model-structure (e.g. neural networks) or are specifically designed to do the same thing (e.g. clustering algorithms) are grouped together. Depending on their complexity, these elements can be further split into sub-elements according to the functional principles underlying the architecture. That is why, for example, neural networks are further split into feedforward neural networks, convolutional neural networks, recurrent neural networks, transformers, spiking neural networks, graph neural networks, Boltzmann machines, multi-network models and competitive learning models.

The concept of the functional unit also informs the choice of the implementation elements in other ways: one could argue that the inclusion of transformers and convolutional neural networks (CNNs), for example, is misplaced, since feedforward neural networks are already part of the taxonomy, and transformers and CNNs are (partially) made up of feed-forward neural networks. However, there are important differences between CNNs, transformers and feedforward networks that impact how each of these architectures work (e.g. the inclusion of the attention mechanism in transformers) and which type of tasks can be implemented with each of these architectures. In that way, a transformer is a functional unit because it implements certain functionalities as part of an AI system (e.g. many kinds of language processing) that a feedforward neural network alone could not do.

Table 4: Elements of the Implementation Category

IMPLEMENTATION	SPECIFICATION
Clustering Algorithm	-
Decision Tree Learning	-
Dimensionality Reduction Algorithm	Linear Technique
	Non-Linear Technique
Kernel Machine	-
Mathematical Optimization Algorithm	Deterministic Approach
	Heuristic Approach
	Trajectory-based
	Evolutionary Algorithms
	Swarm Intelligence
Neural Network-Based	Feedforward Neural Network
	Convolutional Neural Network
	Recurrent Neural Network
	Transformer
	Spiking Neural Network

We do acknowledge that the boundaries between different elements can be fuzzy at times and that the implementation dimension subsumes both architectures (such as neural network-based architectures) and specific algorithms (clustering algorithms, mathematical optimization algorithms etc.). The choice to keep neural networks as a separate entity is due to fact that they are such an important category within modern AI, have a rich inner structure and can be used for many different ends and under different paradigms. For example, neural networks can also be part of reinforcement learning (e.g. Deep Q Networks) or neuro-symbolic techniques (e.g. Logical Neural Networks). In these cases, it is useful to choose both elements for the implementation dimension of the AI system in question.

Moreover, the selection of representative elements for the implementation dimension could be

IMPLEMENTATION	SPECIFICATION
	Graph Neural Network
	Boltzmann Machine
	Multi-Network Model
	Competitive Learning Models
Neuro-Symbolic Techniques	-
Regression Analysis	-
Reinforcement Learning Techniques	Policy Gradient Method
	Actor-Critic Method
	Value-based Method
Symbolic AI Techniques	Reasoning in Knowledge-Based Systems
	Automated Theorem Proving
	Constraint Programming
	Automated Planning
	Rule-based Natural Language Processing
Other NLP	-

complicated further. Each of the elements of the present architecture classification could be split into even more sub-elements, e.g. recurrent neural networks (RNNs) could be split into long-short-term-memory networks, hierarchical RNNs, echo state networks, Hopefield networks and so on; feedforward neural networks could be further divided along regularization techniques or types of loss functions. However, the very purpose of a taxonomy is to find categories and elements under which similar instances can be subsumed and not every little architectural tweak warrants its own element. That is why this classification schema stops at two levels of detail for the architecture category in order to avoid fragmentation due to excessive granularity. Instead, we are confident that the present categories are enough to subsume many other existing architectures. To that end, a list with further examples for each implementation element can be found in the appendix.

The specific selection of implementation elements for this new taxonomy for AI systems is particularly influenced by the understanding of AI systems in the EU AI Act. The boundaries of algorithms and models that belong under the AI umbrella are still fuzzy and few techniques are explicitly excluded from being AI in the AI Act. While there might be no arguments about the validity of certain architectures and algorithms (e.g. neural networks, reinforcement learning techniques etc.) that have become synonymous with AI in recent years, for other algorithms it might not be quite as clear whether they count as AI or not. Examples for this are regression analysis and optimization algorithms: while they do learn from data, they do so in a way that is less complex compared to other AI applications, or in a way that has been long established or that is at least well understood. These cases are included in the current taxonomy as it aims to cover a maximal set of options. The implementation category also makes a point to include older techniques, such as Symbolic AI, as well as very recent developments like spiking neural networks, thus encompassing a broad understanding of AI. While we see this as an asset of our approach, those who only focus on the more contemporary AI methods can simply ignore these older approaches and still work with the same taxonomy.

As with the task and input categories, the implementation category also allows for multiple assignments of elements to the same AI system. An example for this case would be an LLM that takes a prompt and generates an unstructured text response based on the prompt, where the tokenization of the initial input is done with tokenization algorithms and these tokens are then fed into a transformer architecture that generates the output. In this case, the architecture dimension would consist of both 'Other NLP' (for which tokenization algorithms are an example, see table 5) and 'Transformer'.

How the New Taxonomy Works in Practice: Examples

So far, the different categories of the taxonomy have been considered in isolation. However, the purpose of this taxonomy is to be able to classify whole AI systems. To illustrate how this can work in practice, we will look at two examples of varying complexity.

The first example consists of a (physical) ECG device that records and displays the ECG of a patient; this ECG device encompasses a built-in AI system that takes the ECG data as input and automatically detects and classifies anomalies in the ECG like arrhythmias, ST-segment changes, QT prolongation and others. If no anomalies are detected, the ECG is labelled 'healthy'. The label will be displayed alongside the ECG graphs on the output-display of the device. The underlying architecture of the classification algorithm is an xResNet that has been trained using supervised learning. The classification of the AI component of this device under the new taxonomy of AI systems is the following:

Task	Labelling Multi-Class
Input	Time-Series Data
Implementation	Convolutional Neural Network

Since the purpose of the AI system is to assign a label to the incoming ECG data and there are multiple possible labels, its task is clearly a multi-class labelling task. Moreover, the input is the electrical activity coming from the electrodes that are placed on a patient's body. This electrical activity is recorded as voltage data points over time, which makes this a type of time-series data. Lastly, an xResNet is a modern variant of the ResNet (Residual Network) architecture, designed to further improve efficiency and performance in convolutional neural networks. That is why, with regard to the implementation dimension, the xResNet is subsumed under the convolutional neural network (see also a list of examples for individual implementation elements in the appendix).

For a more complex example, consider the case of an AI-assisted job-application tool that is part of a larger HR system. The tool consists of an AI chatbot that, based on the job description, chats with a user as part of a pre-assessment before the actual job interview. Based on the chat with the applicant, this tool provides a summary of its interaction with the candidate and makes a decision about the applicant's suitability for the job. If the candidate is deemed suitable, they get an invitation to a job interview and the summary of the pre-assessment is sent to the recruiter. If they are not deemed suitable, an automatic rejection email is sent to them. The chatbot with all its functionalities is based on a large language model of the GPT-variety. The classification of this tool under the new taxonomy for AI systems is the following:

Task	Generation-Creation-Unstructured Text, Generation-Transformation-Unstructured Text, Labelling Binary
Input	Unstructured Text Structured Data
Implementation	Transformer

The tool carries out multiple functions. Firstly, it chats with the applicant based on the job description. While this chat is partially based on information from the job description, the chatbot generates natural language freely, without limitations to the length of its output and with a high degree of autonomy based on the direct responses of the applicant and a general job interview structure. Therefore, this counts as a generation-creation-unstructured text task. Secondly, it summarizes the conversation with the applicant using its own outputs during the chat interactions and the responses from the candidate. Since a summarization should leave the content of the material intact while shortening its length but keeping it in natural language, this can be considered a generation-transformation-unstructured text task. Lastly, the AI tool also decides about the candidate's suitability for the job. Essentially this amounts to assigning a yes/no label, which is why this is a binary labelling task.

The input to the system consists of the applicant's responses during the chat, which count as unstructured text. Moreover, the information about the job description also feeds into the AI tool. Assuming that the job description in question is just one of many that are stored in the HR system as tabular data, for example, in a relational database, this qualifies as structured data. Lastly, the whole system is based on a generative pre-trained transformer, which is a variant of the transformer architecture.

This example also highlights that the classification is highly dependent on very specific features of the system in question. For instance, that the input about the job description counts as structured data depends on how this data is actually stored in the system. If the AI tool would get access to the job description by parsing the website where the job description is uploaded for the public, it would count as unstructured text instead. These kinds of details are important when attempting to classify an AI application in this new taxonomy for AI systems and cannot be gleaned from a short description. Consequently, it is essential that comprehensive and detailed information about the system is available to the person undertaking the classification.

Discussion: The Question of System Boundaries

As already discussed in the objectives and as demonstrated through the examples, this new taxonomy for AI systems is not singular, i.e. more than one element per category can be assigned to a given AI system. This is explicitly provided for in the task dimension with the concept of composite tasks but can also be observed for the other categories input and implementation. Allowing multiple selections per category reduces the categorization's clarity and definiteness to some extent, but this is counterbalanced by the fact that it enables the concise classification of more complex AI systems. AI technologies develop rapidly and AI applications get more and more complex in the process. Applications that consist of multiple models, handling multiple inputs at once in order to achieve one objective are already reality.

A prevailing question throughout this paper has been the one of system boundaries: while the classification of an AI system under the new taxonomy for AI systems should be unambiguous once it is clear what the AI system does and what the system consists of, the issue here is really the second part. How to define the boundaries of an AI system is an open question, subject to ongoing debate.

Take the example of an AI tool used in an HR system that takes as input the recording of a job interview, transcribes it and gives a recommendation whether the applicant should proceed onto the next round based on an analysis of the transcription. The input is audio data, and the AI system performs both a generation-transformation-document task (the transcription) and a binary labelling task (recommendation whether or not the candidate should proceed to the next round). The question is whether this application should count as one AI system with two tasks, or two AI systems that are subsumed under one AI tool with each having their own classification.

Proponents of the first conception could argue that the two parts would not work as separate systems, since, if we keep the input the same, the second AI system would cease to

function as it needs the transcription from the first part to implement its labelling functionality. The labelling functionality cannot work with only the initial audio data as input. Therefore, the whole system is a functional unit in the sense that both parts are necessary to get from the given input (audio data) to the output of the system (recommendation, i.e. label).

However, one could also argue that the system consists of two functional units, each implementing its own functionality respectively (transcription and recommendation). If the systems both had their proper inputs they could split and function independently on their own; they are not as conceptually related, as, for example, segmentation and identification in the object recognition case described in the task section above. The situation gets even more complicated depending on whether the two functionalities are implemented through different models or through the same model (e.g. a generative pretrained transformer).

The question of system boundaries is a question that this new taxonomy for AI systems cannot answer. Which conception is correct in the case of the HR tool – and many other complex systems like it – depends on the exact definition of both the AI system and the functional unit. For the time being such definitions are still out of reach, which is why the drawing of system boundaries depends heavily on the conceptions of the person responsible for classification, be it the manufacturer, auditor, developer, researcher, or other personnel.

However, in the end it matters less how these boundaries are drawn and more that an explicit decision on system boundaries is made. As explained, the AI System Taxonomy can handle both cases and can give a clear classification of the AI system(s), no matter if it counts as one or two AI systems. This shows that while the question of system boundaries is certainly important for the categorization of individual AI systems, it does not fundamentally undercut the approach of this taxonomy. The principles underpinning the taxonomy prevail regardless of which conception of AI system is chosen in the end.

One might also wonder why the individual categories have internal structures, when the only elements that can actually be assigned to the AI systems are the most detailed ones like 'unstructured text' (in contrast to 'text' as a more general category). This is because this type of structure has value in itself, as it highlights the commonalities between different elements, thereby improving our understanding of that category and AI systems as a whole. Moreover, this structure can also be useful for certain applications of the taxonomy. For instance, for the mapping of AI systems to testing and certification tools, some of these tools might be applicable to all elements of a certain kind (e.g. to all neural network-based architectures). In this case such a tool can be mapped to such an umbrella element with the specification that it applies to all sub-elements, simplifying the mapping and enabling scalable approaches. If this is done across all three main categories, the effort to both classify and select assessment and testing methodologies can be massively reduced.

This also implies that when it comes to the application of the AI System Taxonomy to testing and certification, multiple tools and methods will fall under the same elements, e.g. multiple metrics are available to test, for instance, an AI system that can be categorized as labelling binary – 2D image data – convolutional neural network. In this case, the question which methods are most appropriate to the specific AI system in question is left to the manufacturer or auditor respectively. The taxonomy thus serves as a guide to which testing and certification tools are appropriate for an AI system that has the characteristics captured by the categorization of the AI system under the taxonomy, but due to the myriad of methods available an injective mapping can be considered impossible. It is reasonable to assume that the final choice of appropriate tools must be made in each individual case by the manufacturer and auditor in question based on the detailed specifics of the application.

Lastly, it is also worth noting that the three categories task, input and implementation are not completely independent from each other. In this taxonomy it is theoretically possible that any element from one category can be combined with any element from the other categories, but in practice, some combinations are more common than others. For example, it is highly likely that clustering tasks will be implemented using clustering algorithms or that an AI system implementing a regression task will have structured data input. At the same time, it is almost inconceivable that a generation task will be implemented with a regression algorithm or that a graph neural network will take anything but graph data as input. These limitations are not provided for in this new taxonomy for AI systems, but will instead become apparent in any kind of further application of the taxonomy. Moreover, not explicitly limiting the possible combinations along the three categories allows the taxonomy to stay flexible for edge cases and future developments in the AI landscape.

Outlook

This first version of the new taxonomy for AI systems is a bold step toward bringing clarity and structure to a field that is vast, rapidly evolving, and often conceptually fragmented. Artificial intelligence is advancing at a pace that frequently outstrips our ability to define and categorize it consistently. With this taxonomy, we offer a practical and robust tool that cuts through the complexity and delivers a framework for understanding AI systems in a coherent and systematic way. We firmly believe that it can serve as a valuable asset to all stakeholders in the AI landscape – from developers and researchers to regulators and auditors – by providing a common language and structure that fosters transparency, comparability, and insight.

While the precise delineation of AI system boundaries continues to be debated, this taxonomy sidesteps that uncertainty by remaining flexible and broadly applicable. Any AI system can be meaningfully described through the three core categories we propose: **task**, **input**, and **implementation**. This categorization enables consistent classification regardless of how system boundaries are drawn. Importantly, this is not a static tool. As part of the TÜV AI.Lab AI Assessment Framework, the taxonomy will play a central role in mapping AI systems to appropriate testing and certification tools. It lays the foundation for rigorous, scalable, and meaningful AI assurance – something the field urgently needs.

Appendix

Table 5: Examples for Individual Implementation Elements

IMPLEMENTATION	IMPLEMENTATION SPECIFICATION	EXAMPLE
Clustering Algorithm	-	BIRCH
		Affinity Propagation
		K-Means
		DBSCAN
		OPTICS
		Mean-Shift
		Gaussian Mixture Models
		Fuzzy k-means
		Spectral Clustering
Decision Tree Learning	-	Random Forest
		Gradient-boosted trees
Dimensionality Reduction Algorithm	Linear Technique	Principal Component Analysis
		Linear Discrimination Analysis
		Independent Component Analysis
		Non-negative Matrix Factorization
	Non-Linear Technique	t-distributed Stochastic Neighbourhood Embedding
		Uniform Manifold Approximation and Projection
		Isomap
		Locally Linear Embedding
		Multidimensional Scaling
		Spectral Embedding

IMPLEMENTATION	IMPLEMENTATION SPECIFICATION	EXAMPLE
Kernel Machine	-	Support-Vector Machines
Mathematical Optimization Algorithm	Deterministic Approach	Branch & Bound
		Simplex Method
		DIRECT
	Heuristic Approach	Simulated Annealing
		Hill Climbing
		Beam Search
	Trajectory-Based	Variable Neighbourhood Search
		Greedy Randomized Adaptive Search Procedure
		Tabu Search
	Evolutionary Algorithms	Genetic Algorithm
		Evolutionary Strategies
		Differential Evolution
	Swarm Intelligence	Firefly Algorithm
		Wolf Pack Algorithm
		Particle Swarm Optimization
		Ant Colony Optimization
Neural Network-Based	Feedforward Neural Network	Multilayer Perceptron
		Autoencoder
	Convolution Neural Network	U-Net
		LeNet
		AlexNet
		VGGNet
		ResNet
		DenseNet
		Capsule Neural Network
		R-CNN
	Recurrent Neural Network	LSTM
		Gated Recurrent Unit

IMPLEMENTATION	IMPLEMENTATION SPECIFICATION	EXAMPLE
Neural Network-Based	Recurrent Neural Network	Bidirectional RNN
		Hierarchical RNN
		Attention-based RNN
		Echo State Network
		Continuous-Time RNN
		Hopfield Networks
	Transformer	Encoder-only
		Decoder-only
		Encoder-Decoder
	Spiking Neural Network	Leaky Integrate-and-Fire Networks
	Graph Neural Network	Message Passing Neural Networks
		Temporal Graph Networks
	Boltzmann Machine	Restricted Boltzmann Machines
		Deep Belief Networks
	Multi-Network Model	Diffusion Models
		Generative Adversarial Network
		Siamese Networks
		Adaptive Resonance Theory
	Competitive Learning Model	Learning Vector Quantization
		Kohonen Maps (Self-Organizing Maps)
Neuro-Symbolic Techniques	-	Logic Tensor Network
		Neural Logic Machine
		Logic Boltzmann Machine
		Logic Neural Networks
		Neuro-Symbolic Concept Learner
		Generative Neurosymbolic Machines
Regression Analysis	-	Linear Regression
		Logistic Regression
		Binomial Regression

IMPLEMENTATION	IMPLEMENTATION SPECIFICATION	EXAMPLE
Reinforcement Learning Techniques	Policy Gradient Method	REINFORCE
	Actor-Critic Method	DDPG
		A2C
		TRPO
		PPO
	Value-Based Method	Q-Learning
		Deep Q Networks
		SARSA
Symbolic AI Techniques	Reasoning in Knowledge-Based Systems	Expert Systems
		Fuzzy Controls
	Automated Theorem Proving	Otter
		Vampire
	Constraint Programming	Minion
		CP Optimizer
		RealPaver
	Automated Planning	PPDDL planners
		PANDA
		STRIPS
	Rule-based NLP	spaCy
		NLTK
Other NLP Techniques	-	Tokenization Algorithms
		Word Embeddings

TÜV
AI.LAB

TÜV AI.Lab GmbH
Max-Urich-Str. 3
13355 Berlin
Deutschland
www.tuev-lab.ai
www.tuev-risk-navigator.ai

info@tuev-lab.ai

The TÜV AI.Lab was founded in 2023 as an independent joint venture by the TÜV companies TÜV SÜD, TÜV Rheinland, TÜV NORD, TÜV Hessen and TÜV Thüringen. The TÜV AI.Lab aims to translate the regulatory requirements for AI into practice and make Europe a hotspot for safe and trustworthy AI. To this end, it develops quantifiable conformity criteria and suitable test methods for AI. The AI.Lab also actively supports the development of standards and norms for AI systems.