ETSI GR ENI 018 V2.1.1 (2021-08)

Experiential Networked Intelligence (ENI); Introduction to Artificial Intelligence Mechanisms for Modular Systems

Reference

DGR/ENI-0028_AI_Mechanisms

Keywords

artificial intelligence, cognition, design, software

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - APE 7112B
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° w061004871

Contents

Intellectual Property Rights

Foreword

This Group Report (GR) has been produced by ETSI Industry Specification Group (ISG) Experiential Networked Intelligence (ENI).

Modal verbs terminology

Executive summary

The present document specifies a high-level functional abstraction of the ENI System Architecture in terms of Functional Blocks and External Reference Points. This includes describing how different classes of systems interact with ENI. Processes, models, and detailed information are beyond the scope of the present document.

1 Scope

The purpose of the present document is to provide information on different types of AI mechanisms that can be used for cognitive networking and decision making in modern system design. Bias and ethics will also be addressed. This information can be applied to the ENI reference system architecture (and any other applicable ETSI reports or standards).

2 References

2.1 Normative references

Normative references are not applicable in the present document.

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] ETSI GS ENI 005 (V2.1.1): "Experiential Networked Intelligence (ENI); System Architecture".

[i.2] Cointe, N., Bonnel, G. and Boissier, O.: "Ethical judgment of agents' behaviors in multi-agent systems", AAMAS, pgs 1106-1114, 2016.

[i.3] Anderson, M. and Anderson, S.L.: "GenEth: A general ethical dilemma analyzer", AAAI, pgs 253-261, 2014.

[i.4] Koene, A., Smith, A.L., Egawa, T., Mandalh, S. and Hatada, Y.: "IEEE^{TM} P70xx, Establishing Standards for Ethical Technology", KDD, 2018.

NOTE: Available at http://www.kdd.org/kdd2018/files/project-showcase/KDD18_paper_1743.pdf.

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the terms given in ETSI GS ENI 005 [i.1] and the following apply:

active learning: learning algorithm that can query a user interactively to label data with the desired outputs

NOTE: The algorithm proactively selects the subset of examples to be labeled next from the pool of unlabeled data. The idea is that an ML algorithm could potentially reach a higher level of accuracy while using a smaller number of training labels if it were allowed to choose the data it wants to learn from.

batch learning: type of offline learning algorithm that is updated (i.e. retrained) periodically

catastrophic forgetting: tendency of an artificial neural network to forget previously learned information when learning new information

concept drift: not taking changing data and its meanings into account when training an ML model

one-cold vector: 1 × N matrix (vector) used to distinguish each word in a vocabulary from every other word in the vocabulary, where the vector consists of 1s in all cells with the exception of a single 0 in a cell used uniquely to identify the word

one-hot vector: 1 × N matrix (vector) used to distinguish each word in a vocabulary from every other word in the vocabulary, where the vector consists of 0s in all cells with the exception of a single 1 in a cell used uniquely to identify the word

3.2 Symbols

Void.

3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

AI	Artificial Intelligence
ANN	Artificial Neural Network
BERT	Bidirectional Encoder Representations from Transformers
GPT	Generative Pre-trained Transformer
LSTM	Long Short-Term Memory
MIT	Massachusetts Institute of Technology
ML	Machine Learning
NLP	Natural Language Processing

4 Artificial Intelligence for Modular Systems

4.1 Introduction

Machine learning algorithms learn a solution to a problem from sample data.

Historically, machine learning has focused on non-incremental, offline learning tasks (i.e. where the training set can be constructed a priori and training stops once this set has been duly processed). There are, however, a number of areas, such as agent-based learning and processing sequential data, where learning tasks are inherently incremental.

Machine learning systems that use offline learning do not change their approximation of the target function when the initial training phase has been completed unless the system can detect that the input data has changed significantly. In contrast, machine learning systems that use online learning continuously re-evaluate their target function. The advantage of offline learning is that training is computationally intensive. The advantage of online learning is that if small changes, such as trends, are expected, online training will perform better than offline variants.

The machine learning process adjusts parameters to minimize observed errors; this does not mean that the error rate reaches 0. However, it does mean that if the error rate becomes too high, then the Artificial Neural Network (ANN) needs to be redesigned. This in turn is done by defining a function that minimizes the difference between the observed and desired values.

4.2 Types of Learning

4.2.1 Introduction

In each of the following clauses, each type of algorithm has a large number of variants available. No single learning algorithm works in an optimal manner for all problems.

4.2.2 Supervised

Supervised learning defines a function that maps an input to an output based on example pairs of labelled inputs and outputs. Each input is a tuple that consists of an input object and a desired output value. The learning function analyses the training data and produces a function that can determine the class labels for new data.

There are a large number of algorithms available. No single supervised learning algorithm works in an optimal manner for all problems.

4.2.3 Semi-Supervised

Semi-supervised learning is a hybridisation of supervised and unsupervised learning, where the training data consists of both labelled and unlabelled data.

4.2.4 Unsupervised

Unsupervised learning defines a function that maps an input to an output without the benefit of the data being classified or labelled. The input data can be modelled as probability densities.

4.2.5 Reinforcement Learning

Reinforcement learning uses software agents to take actions in an environment in order to maximize a cumulative reward. In this approach, the learning agent is not told which actions to take, but instead is responsible for discovering which actions yield the highest reward.

4.2.6 Feature Learning

Feature learning analyses raw input data to learn the most important characteristics and behaviour representations of those data that make it easier to discover information from raw data when building different types of predictors (e.g. classifiers).

4.2.7 Rule-Based Learning

Rule-based learning uses learned rules to represent the knowledge of a system. This type of system learns rules to make decisions, instead of using a model. These rules are different than other types of rule-based systems because this set of rules are learned, while rules in other types of systems are defined. The rules are typically imperative rules.

4.2.8 Explanation-Based Learning

Explanation-based learning uses an explanation-driven approach that enables a search procedure, constrained by general domain knowledge related to the context of the actual problem, to be used to provide more accurate and efficient learning in knowledge-intensive systems.

4.2.9 Federated Learning

Federated learning is an approach that trains a centralized model using decentralized data that is distributed across multiple entities holding local data samples, without exchanging their data samples. Each device trains the model on their own local data set, and then each client sends a model or model update to a centralized service, which aggregates each client's contribution into one global model. The centralized service then distributes the global model back to the clients.

4.2.10 Active Learning

Active learning is an iterative supervised learning algorithm where the algorithm can actively query an oracle (e.g. a human annotator) to obtain the correct label. This approach enables the learning algorithm to interactively choose the data it will learn from. Active learning iteratively selects the most informative examples to acquire their labels and trains a classifier from the updated training set, which is augmented with the newly selected examples. Unlike conventional supervised learning, it permits a learning model to evolve and adapt to new data. Active learning is concerned with learning accurate classifiers by choosing which examples will be labelled, reducing the labelling effort and the cost of training an accurate model. Active learning is appropriate for machine learning applications where labeled data is costly to obtain but unlabeled data is abundant.Active learning is especially important where objects can have multiple labels that belong to various categories (e.g. a network device has multiple roles, or an image can be labelled as containing both mountains, beach, and ocean). The main challenge in performing.

4.3 Model Training

4.3.1 Online Model Training

A dynamic model is a model that is trained online. That is, data is continually ingested and is incorporated into the model through continuous updates. The motivation is that most information sources change, and if the model is not trained on the data changes from its information sources, it becomes stale and out-of-date. Worse, it will likely produce inaccurate, or even incorrect, predictions.

Hence, online machine learning ingests data sequentially; the ingested data is then used to update the best predictor for future data at each step. This is fundamentally different from offline or batch learning algorithms that generate the best predictor by learning on the entire training data set at once. Online learning is a common technique used in areas of machine learning where it is computationally infeasible to train over the entire dataset, requiring the need of out-of-core algorithms. It is also used in situations where it is necessary for the algorithm to dynamically adapt to new patterns in the data, or when the data itself is generated as a function of time, e.g. stock price prediction. Online learning algorithms may be prone to catastrophic forgetting, which can be addressed by using incremental learning in addition to online learning.

4.3.2 Offline Model Training

Offline model training is the use of a static model that is trained once; that trained model is then used for a period of time. The term batch learning is used when an offline model is updated (i.e. retrained) periodically. Batch learning is a compromise between online and offline model training, and seeks to keep the algorithm updated by scheduling training updates. In either case, the learned models are trained once, and then updated as needed.

4.3.3 Comparison

Static models are easier to build and test, but cannot adapt to changing information sources. Static models still need updating on a regular basis; otherwise, the data that the model is trained on because stale and out-of-date.

In contrast, online learning is better suited for those problems where samples are presented over time and where the probability distribution of samples is expected to also change over time.

Online learning models are harder to be managed in production level stage after deployment. Since the model has to churn large amounts of the dataset in real-time, any changes in the pattern of data - or more commonly known as concept drift - will affect the overall performance and prediction of the data. In contrast, Offline learning, however, with the model being constant after the deployment stage (with same types and pattern of data), it is easier to maintain the whole network or cluster with minimal supervision and control.

Table 1: Comparison of Online vs. Offline Learning

Features	Online Learning	Offline Learning
Model Complexity	Complexity varies dynamically as a function of changing data	Complexity is much less since the model training is constant
Update Period	Model is sequentially trained on one instance at a time	Model is trained after consuming the entire batch of new data
Computational Power Required	More computations since model is continuously refined	Fewer and more efficient computations since they are scheduled periodically
Implementation	Harder to implement and manage	Easier to implement and manage
Usage	Applications where new data and patterns are frequently emerging (e.g. networking, health, finance)	Applications where data changes very slowly (if at all), such as image classification
Potential Problems	Prone to *catastrophic forgetting* (i.e. forgetting previously learned information when new information is learned)	The predictor is prone to *concept drift* (i.e. not taking changing data and its meanings into account)

4.4 Bias

4.4.1 Definition

Bias is defined as "the systematic difference in treatment of certain objects, ideas, or people in comparison to others". In this definition, *systematic* means that the difference in treatment is predictable and typically constant.

4.4.2 Types of Bias

4.4.2.1 Algorithmic Bias

Algorithmic bias is defined as "an algorithm that possesses systematic and repeatable errors that create unfair outcomes". In this definition, "algorithmic" means the combination and interaction of the algorithm with data that it uses and outputs that it provides. Algorithmic bias may exist even when there is no intention by the algorithm developer to discriminate. For example, the data set may contain historical discrimination. A more subtle issue is when the data set contains correlations between sensitive privacy attributes.

In the fields of statistics and artificial intelligence, bias typically has five causes:

- Variables might not be independently identically distributed.
- Errors might be concentrated in a particular class in the data set.
- Unfair sampling (e.g. incorrect collection or selection of objects from a statistical population).
- Unfair training (e.g. training set does not contain a diverse set of data).
- Context-specific implementation problems.

While hard to detect, the first four types of bias can be fixed using a set of processes developed specifically for each problem. The fifth source of bias is context-specific, and consists of technical and emergent bias.

4.4.2.2 Technical Bias

Technical bias is an epistemological problem. Put another way, when developers translate constructs into a form that computers can process, the translation process itself may introduce bias. Hence, understanding how features are engineered and performed and processed, and where errors can be introduced, is critical for avoiding technical bias.

There are four types of technical bias. Labeling bias occurs when the label assigned to the training data does not adequately represent the underlying semantics of that outcome. Measurement bias is introduced when an inappropriate measurement scale is used (e.g. changing from an ordinal to a nominal scale), since this affects both the sample and how the consumer using the sample is represented. Modeling bias occurs when features are introduced that are not truly representative of the population under study and, more importantly, are predictive of the outcome. This can be viewed as a compression problem in which high-dimensional data that interact in multiple ways are represented by lower-dimensional data that interact in more limited ways. Optimization bias is introduced when the model parameters are tuned to achieve certain objectives.

4.4.2.3 Inductive Bias

The inductive bias of a learning algorithm is defined by how the algorithm chooses among all models that fit the data being examined equally well. The underlying problem is how to choose a learner's hypothesis space so that it is large enough to contain a solution to the problem being learnt, yet small enough to ensure reliable generalization from reasonably-sized training sets. Learning from examples thus has an inherent bias that determines which model is best, even though the instances have not yet been observed by the learner. Hence, the goal is to ensure that the learning program has a desirable and productive inductive bias.

4.4.2.4 Emergent Bias

Emergent bias occurs once the ML model is in use. It is caused by such factors as:

- the introduction of new forms and types of data that were not considered and trained by the existing algorithm;
- a mismatch between the original purpose of the ML system and the (new) application that is using it;
- a change in the underlying environment that the ML system is being used on (e.g. the introduction of COVID-19 and its effects on retail prediction systems).

4.5 Ethics and Ethical Decision-Making

4.5.1 Introduction

There are many definitions of ethics. One definition [i.2] defines ethics as a normative philosophical discipline of how a person or object should act towards others. It comprises three dimensions:

1) Consequentialist Ethics: an agent is ethical if and only if it considers the consequences of each decision and chooses the decision that has the most moral outcome.

2) Deontological Ethics: an agent is ethical if and only if it respects obligations, duties, and rights appropriate for a given situation.

3) Virtue Ethics: an agent is ethical if and only if it acts according to a set of moral values.

4.5.2 Definitions

An *ethical dilemma* is a situation in which any available decision leads to infringing one or more ethical principles.

4.5.3 Embedding Ethical Decision-Making into AI Systems

An approach to embedding ethics into AI systems is based on the above four notions. The hardest is to translate ethical dilemmas to a form that a computer is able to understand. Ethical issues have important, social, political, and legal connotations that typically are not part of the skill set of an AI developer. Consequently, software tools are used to detecting and removing ethics in AI systems. Two types of software tools have emerged:

1) those based on expert review of proposed decisions; and

2) those based on crowdsourcing mechanisms.

Examples include [i.3] and the Moral Machine project of MIT. The MoralMachine project is described in https://www.media.mit.edu/projects/moral-machine/overview/, and its implementation is at https://www.moralmachine.net.

4.5.4 Existing Work in Standards and Fora

There are several important initiatives concerned with ethical decision-making, and more specifically, how to help ensure that AI systems are both prioritized and implemented. Three of these are described in this clause.

The mission of the The IEEE Global Initiative on Ethics of Autonomous and Intelligent Systems is *"to ensure every stakeholder involved in the design and development of autonomous and intelligent systems is educated, trained, and empowered to prioritize ethical considerations so that these technologies are advanced for the benefit of humanity"*. The Ethics Certification Program for Autonomous and Intelligent Systems (ECPAIS) is to create specifications for certification processe that advance transparency, accountability and reduction in algorithmic bias in Autonomous and Intelligent Systems.

The IEEE^{TM} P7000 series of Standards (P7000-P7014) [i.4] are a set of thirteen working groups to create the IEEE^{TM} P70xx series ethics standards, and associated certification programs, for Intelligent and Autonomous systems. These standards are:

IEEE^{TM} P7000:	Model Process for Addressing Ethical Concerns During System Design.
IEEE^{TM} P7001:	Transparency of Autonomous Systems.
IEEE^{TM} P7002:	Data Privacy Process.
IEEE^{TM} P7003:	Algorithmic Bias Considerations.
IEEE^{TM} P7004:	Standard on Child and Student Data Governance.
IEEE^{TM} P7005:	Standard on Employer Data Governance.
IEEE^{TM} P7006:	Standard on Personal Data AI Agent Working Group.
IEEE^{TM} P7007:	Ontological Standard for Ethically Driven Robotics and Automation Systems.
IEEE^{TM} P7008:	Standard for Ethically Driven Nudging for Robotic, Intelligent and Autonomous Systems.
IEEE^{TM} P7009:	Standard for Fail-Safe Design of Autonomous and Semi-Autonomous Systems.
IEEE^{TM} P7010:	Wellbeing Metrics Standard for Ethical Artificial Intelligence and Autonomous Systems.
IEEE^{TM} P7011:	Standard for the Process of Identifying and Rating the Trustworthiness of News Sources.
IEEE^{TM} P7012:	Standard for Machine Readable Personal Privacy Terms.
IEEE^{TM} P7013:	IEEE Draft Inclusion and Application Standards for Automated Facial Analysis Technology.
IEEE^{TM} P7014:	Standard for Ethical considerations in Emulated Empathy in Autonomous and Intelligent Systems.

4.6 Natural Language Processing using AI: an Overview

4.6.1 Introduction

Natural Language Processing (NLP) is a branch of machine learning that enables machines to "understand" human language. A combination of linguistics, computer science, and machine learning, NLP works to transform regular spoken or written language into a form that can be processed by machines.

NLP is a challenging task. An NLP program needs to be able to understand:

- morphology (the structure of words);
- syntax (the way words are used to form phrases and sentences);

- lexical semantics (the meaning of individual words);
- compositional semantics (the meaning of a phrase or sentence based on syntax);
- pragmatics (meaning in context).

Most natural languages are inherently ambiguous. Here are examples of various types of ambiguity:

lexical:	is the word "bank" a noun or a verb, and for either, which meaning is correct (e.g. river bank vs. bank (financial institution) vs. bank, as in rely).
syntax:	prepositional phrases can be associated with different parts of a sentence (e.g. "John programmed the device on the rack with a PC" could mean "John, using a PC, programmed the device on the rack", or "John programmed a device on the rack which had a PC in it" or "John, on the rack and using a PC, programmed the device").
referential:	pronouns can be ambiguous.

This is why *controlled languages* (i.e. a restricted version of a single Natural Language that uses a subset of the grammar of the Natural Language) *are recommended* in many NLP applications, including the ENI System Architecture.

The next set of clauses provide a brief introduction to several popular methods of NLP.

4.6.2 Embeddings

Originally, words were represented as discrete symbols, which can be represented by one-hot vectors. One-hot vectors ensure that the algorithm does not assume that higher numbers are more important. Put another way, a larger value does not make that value "more important" than a smaller value. This is especially important for natural language processing, since verbs and participles are typically longer than the root word in English.

A vector's dimension is the number of words in entire vocabulary. The problem with words as discrete symbols is that there is no natural notion of similarity for one-hot vectors. Thus, the alternative is to learn to encode similarity in the vectors themselves. The core idea is that a word's meaning is given by the words that frequently appear near it.

In contrast, Text Embeddings are real-valued vector representations of strings, where instead of a one-hot vector, a dense vector is built for each word, chosen so that it is similar to vectors of words that appear in similar contexts. This enables deep learning to be effective on smaller datasets, as they are often the first inputs to a deep learning architecture and the most popular way of transfer learning in NLP. Text embeddings identify the hidden patterns in word co-occurrence statistics of language corpora (i.e. a collection of documents that the NLP is trained on), which include grammatical and semantic information. Two examples of Text Embeddings are Word2Vec and GloVe.

Word2Vec uses a large corpus of text in which every word in a fixed vocabulary is represented by a vector. For each position τ in the text, which has a center word π and context words the similarity of the word vectors for π and σ are used to calculate the probability of σ given cπ (or vice versa). The word vectors are adjusted to maximize this probability. The elimination of high frequency (e.g. meaningless) words from the dataset (e.g. articles and prepositions) helps improve model accuracy and training time. Additionally, negative sampling can be used for every input by updating the weights for all the correct labels, but only on a small number of incorrect labels. A problem with this type of model is that it is window-based, meaning the co-occurrence statistics of the corpus are not used efficiently, resulting in suboptimal embeddings. The GloVe model seeks to solve this problem by capturing the meaning of one word embedding with the structure of the whole observed corpus. The model trains on global co-occurrence counts of words and makes a sufficient use of statistics by minimizing least-squares error and, as a result, produces a word vector space with meaningful substructure. Such an outline sufficiently preserves words' similarities with vector distance.

4.6.3 Long Short-Term Memory Models

Long Short-Term Memory (LSTM) networks introduce gates and an explicitly defined memory cell. Each neuron has a memory cell and three gates: input, output and forget. The function of these gates is to safeguard the information by stopping or allowing the flow of it. The input gate determines how much of the information from the previous layer gets stored in the cell, while the output layer determines how much of the next layer gets to know about the state of this cell. The forget gate determines which characters are forgotten for the next layer of processing. LSTMs are currently the default model for most sequence labeling tasks.

4.6.4 Attention

The attention mechanism enables a neural network to dynamically highlight relevant features of the input data (e.g. a sequence of textual elements). The core idea behind attention is to compute a weight distribution on the input sequence, assigning higher values to more relevant elements. Attention enables the relevance of the input elements to be estimated, and also enables these elements to be combined into a compact vector representation that condenses the characteristics of the most relevant elements. Attention does not try to encode the full source sentence into a fixed-length vector; instead, it enables the decoder to focus on different parts of the source sentence at each step of the output generation. This enables the model to learn what to focus on based on the input sentence and what it has produced so far. Because this vector is smaller than the original input, it requires fewer computational resources to be processed at later stages, yielding a computational gain. Attention can be applied in different dimensions, such as space, time, or even semantics.

4.6.5 Transformer Models

4.6.5.1 Introduction

A Transformer is a deep learning model that utilizes attention, weighing the influence of different parts of the input data. The Transformer is the first transduction (i.e. convert input sequences into output sequences) model relying entirely on self-attention to compute representations of its input and output without using sequence-aligned RNNs or convolution. Transformers are designed to handle sequential input data, such as natural language, but do not require that the sequential data be processed in order. Rather, the attention operation identifies context for any position in the input sequence. This enables its implementation to be inherently parallel.

4.6.5.2 BERT Models

Bidirectional Encoder Representations from Transformers (BERT) are a set of models that apply bidirectional training of an attention model, called a Transformer, to model the language. Research has shown that a language model which is bidirectionally trained can have a deeper sense of language context and flow than single-direction language models.

NOTE: This is for further study in Release 3 of the present document.

4.6.5.3 GPT Models

Generative Pre-Trained Tranformer are a family of models that do not used supervised learning. Rather, they use semi-supervised learning: unsupervised learning on unlabelled data, followed by supervised learning for fine-tuning the model by providing examples of specific tasks to be performed, such as classification or textual entailment.

NOTE: This is for further study in Release 3 of the present document.

4.6.5.4 Sparse Transformers

The attention mechanisms for BERT, GPT, and other similar models provide high accuracy, but are difficult to implement efficiently on GPUs and CPUs, due to the complex data movement and lack of arithmetic operations. Sparse transformers reduce the attention computation and memory access required, and take advantage of inherent redundancy in a human language.

NOTE: This is for further study in Release 3 of the present document.

4.6.5.5 Mixture of Experts Model

Mixture of Experts is an ensemble learning technique that was original developed for neural networks. It involves decomposing predictive modeling tasks into sub-tasks, training an expert model on each, developing a gating model that learns which expert to trust based on the input to be predicted, and combines the predictions. For example, NLP tasks can assign different experts to handle different parts of speech or grammatical rules.

NOTE: This is for further study in Release 3 of the present document.

5 Summary and Recommendations

The present document has provided an introduction for how artificial intelligence is used in the ENI System Architecture. Different types of learning and model training were defined, and a generic overview of bias and ethical decision-making was given. The present document concluded with an introduction to natural language processing principles.

These concepts are all used in the design of ETSI GS ENI 005 [i.1] and are applicable to other ETSI reports and standards.

Thus, the present document recommends that the contents of the present document are applicable as primary concepts that sustain ETSI GS ENI 005 [i.1] and other related documents and specifications.

History

Document history		
V2.1.1	August 2021	Publication