International Journal of Law and Information Technology, 2023, **31**, 283–301
https://doi.org/10.1093/ijlit/eaad026
Advance access publication 15 December 2023
Article

AI Regulation and the Protection of Source Code*

Andrew D. Mitchell,† Dominic Let,‡ Lingxi Tang§

ABSTRACT

Artificial intelligence (AI) has been rapidly developing since the mid-20th century. Today, AI is deployed in many aspects of our daily lives and is increasingly ubiquitous. At the same time, concerns about AI, including discrimination, privacy, and security, have prompted calls for greater regulation. To this end, regulators may seek access to AI's source code (the human-readable program for the software). However, many trade agreements contain provisions prohibiting source code disclosure as a condition for market access in the relevant countries. This article considers the tension between AI regulation and these provisions. In situations where such provisions apply, for AI regulation to remain compliant, we argue that regulators can only require access to source code in exceptional cases where the general exceptions in the General Agreement on Trade in Services apply.

KEYWORDS: artificial intelligence (AI); general agreement on trade in services (GATS); Source Code; EU AI Act.

INTRODUCTION

In 1997, IBM's Deep Blue, a supercomputer powered by artificial intelligence ("AI"), achieved a symbolic victory over World chess champion Garry Kasparov.[1] This significant win (or defeat) cast AI into the limelight, demonstrating the potential for AI to beat humans in particular activities. Today, modern chess engines such as Stockfish and AlphaZero have advanced to a point where it is practically impossible for any human to come even close to beating a computer at chess.[2]

* The opinions expressed in this article are those of the authors and do not necessarily reflect the views of their employers. The authors would like to thank Chris Marsden, Neha Mishra, Piero Guasta and Theodore Samlidis for their helpful comments on an earlier draft.

† Andrew D. Mitchell, Professor, Associate Dean (Research), Faculty of Law, Monash University; PhD (Cantab), LLM (Harv), LLB(Hons), BCom(Hons) (Melb); orcid.org/0000-0001-8399-8563.

‡ Dominic Let; LLB (NUS), MComp CS Candidate (NUS).

§ Lingxi Tang; MEng (Oxon).

[1] Joanna Goodrich, 'How IBM's Deep Blue Beat World Champion Chess Player Garry Kasparov' (*IEEE Spectrum*, 25 January 2021) <https://spectrum.ieee.org/how-ibms-deep-blue-beat-world-champion-chess-player-garry-kasparov> accessed 7 August 2022.

[2] Robert Siegel, '20 Years Later, Humans Still No Match for Computers on the Chessboard' (*npr*, 24 October 2016) <https://www.npr.org/sections/alltechconsidered/2016/10/24/499162905/20-years-later-humans-still-no-match-for-computers-on-the-chessboard> accessed 9 August 2022.

We may understand AI as an algorithm capable of making decisions that mimic human intelligence. AI technology today allows us to build AI systems capable of making decisions in specific scenarios. Making chess decisions is one of many scenarios where AI systems have been developed. This article focuses on one of the most popular subsets of AI, machine learning, which will be further discussed in Part III below.

Rapid advancements in AI come with considerable risks, some of which have already manifested. Immediate AI dangers range from accidents caused by driverless cars[3] to more insidious threats, such as political polarization and manipulated elections.[4] Given AI's risks, it is crucial for laws and regulations specific to AI to be developed to ensure that AI continues to be used responsibly.

Inspecting an AI system's source code is often essential to assess its safety. However, many international trade agreements contain provisions that prevent governments from making access to or the transfer of source code a condition for market access ("Source Code Provisions"). Firms that sell software or operate on digital platforms, including AI firms, will often invest significant resources in developing the source code underlying their products. Hence, protecting their source code would be paramount to such companies. Any requirement to disclose it, even to regulators, could be tantamount to demanding a company expose its trade secrets.

As such, there is a direct tension between trade agreements and AI regulation, should the relevant AI regulation mandate source code disclosure. On one hand, there are good reasons for trade agreements to prohibit source code disclosure as a condition for market access, primarily to promote technological trade between countries. Parties to such a trade agreement should uphold their obligations to continue enjoying its benefits. On the other hand, source code review is crucial for effective and comprehensive AI testing, without which, regulators would be working with their hands tied.

This article explores the avenues for regulators when formulating an AI regulatory regime considering Source Code Provisions. In particular, it seeks to determine how regulators may ensure AI regulatory regimes comply with Source Code Provisions and where they might strike a balance between regulation and promoting innovation.

Given that the EU draft AI regulation ("Draft AI Act")[5] is the world's first concrete proposal for regulating AI, it is likely to influence subsequent AI laws. As such, a significant part of this article analyses the Draft AI Act, primarily its source code disclosure obligations. Nevertheless, it should be noted that the main point of the article is not to determine if the Draft AI Act is compliant with Source Code Provisions (although this will be discussed in Part V). Rather, this article analyses AI regulation and Source Code Provisions in general, and it is hoped that through a discussion of the Draft AI Act and the EU's obligations under Source Code Provisions, this article highlights important points that would be beneficial for future developments of AI law.

At the time of writing, the EU Council had released its general approach[6] (dated 25 November 2022) for the Draft AI Act ('General Approach'), which saw significant changes to some of the provisions concerning source code disclosure since the first proposal published by the EU Commission. The European Parliament had also adopted and introduced its own

[3] Bryan Pietsch, '2 Killed in Driverless Tesla Car Crash, Officials Say' (*The New York Times*, 18 April 2021) <https://www.nytimes.com/2021/04/18/business/tesla-fatal-crash-texas.html> accessed 9 August 2022.

[4] 'Artificial intelligence: threats and opportunities' (*European Parliament – News*, 2 May 2022) <https://www.europarl.europa.eu/news/en/headlines/society/20200918STO87404/artificial-intelligence-threats-and-opportunities> accessed 7 August 2022.

[5] Proposal for a Regulation of the European Parliament and of the Council Laying Down Harmonized Rules on Artificial Intelligence (Artificial Intelligence Act) and Amending Certain Union Legislative Acts [2021] SEC(2021) 167 final (EU Commission's Proposal).

[6] Proposal for a Regulation of the European Parliament and of the Council laying down harmonised rules on artificial intelligence (Artificial Intelligence Act) and amending certain Union legislative acts – General approach 14954/22 (General Approach).

amendments[7] to the EU Commission's proposal on 14 June 2023 ('Parliament's Position'). This article considers both proposals, as well as the first proposal published by the EU Commission ('EU Commission's Proposal'). To avoid confusion, for this article, 'Draft AI Act' refers to the proposals collectively. Where there is a need to refer to a particular proposal, we will do so specifically.

In this article, we suggest that the scope of many Source Code Provisions is unclear, given that there has been no attempt to define 'source code' in trade agreements. To remedy this, this article proposes that clarifications be made to the ambit of protection of Source Code Provisions. This article addresses the uncertainty surrounding application programming interfaces ('APIs'), as it is often unclear whether APIs fall under the meaning of 'source code'.

We also suggest that the Draft AI Act may be compatible with the source code discipline, although it allows regulators to require source code disclosure where high-risk AI systems[8] are concerned. This is because the general exceptions found in the General Agreement on Trade in Services[9] ("GATS") may apply to allow regulators to require source code disclosure without breaching the EU's obligations under the Source Code Provisions.

Lastly, based on the approach of the Draft AI Act, we suggest that AI regulations apply different testing methods depending on the type of AI system concerned. Some AI systems may be tested without access to source code. While this is not relevant for the Draft AI Act (since it only applies to high-risk AI systems), it could be applicable to future AI regulations should they implement obligations for AI systems across the board.

This article proceeds as follows. Part I gives an overview of AI systems, highlighting the issues that result in regulatory difficulty. Part II provides a brief background of Source Code Provisions and their scope of protection. Part III enumerates the various methods by which regulators may undertake testing or compliance assessment of AI systems, and analyses their compliance with source code disciplines in Part IV. Finally, Part V discusses the Draft AI Act, focusing on the provisions that are incompatible with the source code discipline, and Part VI concludes.

AN INTRODUCTION TO ARTIFICIAL INTELLIGENCE

Broadly, we may understand AI as machines that demonstrate some form of 'intelligence'. Specifically, the definition and ambit of AI is difficult to pin down, given that it first requires us to define 'intelligence', which is controversial in itself. The EU Commission's Proposal defines 'AI system' as 'software that is developed with one or more of the techniques and approaches listed in Annex I and can, for a given set of human-defined objectives, generate outputs such as content, predictions, recommendations, or decisions influencing the environments they interact with'.[10] On the other hand, the Parliament's Position chose to define AI broadly, as 'a machine-based system that is designed to operate with varying levels of autonomy and that can, for explicit or implicit objectives, generate outputs such as predictions, recommendations, or decisions, that influence physical or virtual environments'.[11] It is likely that the Parliament's position adopted a broader definition of AI in order to prevent the definition from being overtaken by fast-paced developments in the field of AI.

[7] Amendments adopted by the European Parliament on 14 June 2023 on the proposal for a regulation of the European Parliament and of the Council on laying down harmonized rules on artificial intelligence (Artificial Intelligence Act) and amending certain Union legislative acts P9_TA(2023)0236 (Parliament's Position).

[8] Within the meaning of Art 6 of the Draft AI Act.

[9] GATS: General Agreement on Trade in Services, Apr. 15, 1994, Marrakesh Agreement Establishing the World Trade Organization, Annex 1B, 1869 U.N.T.S. 183, 33 I.L.M. 1167 (1994) (GATS).

[10] EU Commission's Proposal (n 5), Art 3.

[11] Parliament's Position (n 7), Amendment 165.

Nevertheless, it is not the focus of this article to discuss the exact definition of AI. Rather, for the sake of simplicity, this article focuses on machine learning ('ML'), which is an uncontroversial subset of AI systems. This is also reflected in Annex I para (a) of the Draft AI Act, which expressly includes '[m]achine learning approaches, including supervised, unsupervised and reinforcement learning" in the definition of an "AI system'.

ML systems are pervasive in almost all aspects of society.[12] Most notably, in the past year from the time of writing, large language models ('LLMs') were cast into the limelight by the overwhelming popularity of ChatGPT. ChatGPT, the LLM released by OpenAI, reached an estimated 100 million monthly users in its first two months of release.[13] Importantly, it highlighted the capability of AI and its significant potential for widespread commercial use. This calls for prompt AI regulation, now more than ever, to mitigate the potential risks that such ML models might bring. Hence, given that ML systems are likely the most popular and high-risk forms of AI till date, this article will primarily discuss such systems. This article's future references to AI systems refer to ML systems.

At this juncture, it is helpful to describe how an ML system typically functions.

Figure 1 shows the simplified design architecture of a typical ML algorithm. Firstly, a training dataset with data samples, each labelled with the appropriate outcome/decision or prediction, is used to 'train' the ML model. The algorithm 'trains' the model by having it detect recurring patterns in each data sample, such that it 'learns' the relationship between the input data and output prediction, thus the term 'machine learning'. The resultant model can be simply

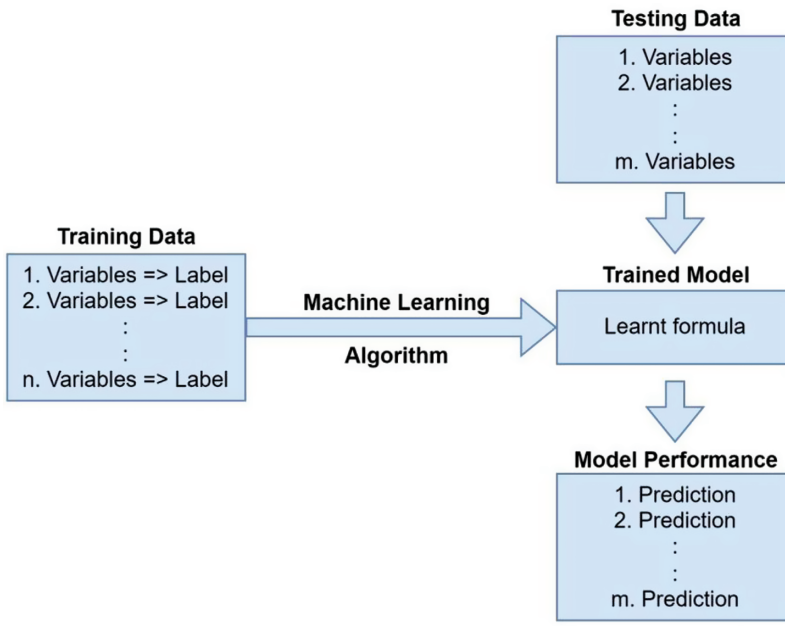

Figure 1. Design architecture of an ML algorithm

[12] From consumer technology, such as self-driving cars, to finance, such as commodity price prediction models, to the arts, such as image generation models.

[13] Gerrit De Vynck, 'ChatGPT Loses Users for First Time, Shaking Faith in AI Revolution' (*The Washington Post*, 7 July 2023) <https://www.washingtonpost.com/technology/2023/07/07/chatgpt-users-decline-future-ai-openai/> accessed 3 September 2023.

understood as a formula that defines the relationship between input data and output prediction, where the formula parameters are 'learnt' via data exposure. For example, if an ML engineer would like to train a model to discern between chihuahuas and muffins in images, they would provide images of chihuahuas (labelled as chihuahuas) and pictures of muffins (labelled as muffins) to the ML algorithm.[14] In this example, the resultant model could take an image as input and decide whether the image depicts a chihuahua or a muffin. The performance of the trained model can then be tested using a testing dataset, consisting of unique data samples usually not found in the training dataset. Thus, the main factor distinguishing AI models is the different training dataset and learning algorithms[15] used.

Much of the difficulty in regulating ML systems comes from what has been described as its 'opacity'. This refers to the fact that it is difficult, especially in complex systems, for a human to predict the output of an ML model definitively. As highlighted by Burrell,[16] '[m]achine learning models that prove useful … possess a degree of unavoidable complexity', such that 'the workings of machine learning algorithms can escape full understanding and interpretation by humans, even for those with specialized training, even for computer scientists'. Doshi-Velez et al described ML model interpretability as 'the ability to explain or to present in understandable terms to a human'.[17] The Deep Patient[18] model is an example of an 'opaque' ML model. Trained using data from about 700,000 individuals, the model proved extremely effective at predicting diseases in patients, even those that are challenging to predict by expert physicians, such as schizophrenia. However, how the system works remains an enigma, as even the research lead has been unable to explain the model's inner mechanisms.[19]

The 'opacity' of ML models is problematic for regulators as it is often not possible to fully understand how a model makes its decisions. This makes it challenging to ensure that a commercially deployed ML system will perform exactly as it should without making any unlawful or undesirable decisions.

THE BACKGROUND OF 'SOURCE CODE PROVISIONS'

We pause the discussion on AI to give an overview of Source Code Provisions.

'Source code'

'Source code' refers to instructions a programmer supplies to the computer, typically to perform specific tasks. It generally refers to high-level languages consisting of human-readable alphanumeric characters, in contrast with object code.[20] This qualification is vital as the high-level programming language supplied by the programmer typically undergoes several stages of compilation before being executed by the computer. For the C programming language, source code written by the programmer is first compiled into assembly, which is specific to the machine's

[14] Mariya Yao, 'Chihuahua or Muffin? My Search for the Best Computer Vision API' (*freeCodeCamp*, 12 October 2017), <https://www.freecodecamp.org/news/chihuahua-or-muffin-my-search-for-the-best-computer-vision-api-cbda4d6b425d/> accessed 9 August 2022.

[15] Examples of such learning algorithms include regression, neural networks, support vector machines, etc.

[16] Jenna Burrell, 'How the machine 'thinks': Understanding opacity in machine learning algorithms' (2016) January–June 2016 Big Data & Society, 1–12, 5.

[17] Sciforce, 'Introduction to the White-Box AI: the Concept of Interpretability' (*Medium*, 31 Jan 2020) <https://medium.com/sciforce/introduction-to-the-white-box-ai-the-concept-of-interpretability-5a31e1058611> accessed 07 August 2022.

[18] Riccardo Miotto and others, 'Deep Patient: An Unsupervised Representation to Predict the Future of Patients from the Electronic Health Records' (2016) 6 Sci Rep 26094. <https://www.nature.com/articles/srep26094> accessed 7 August 2022.

[19] Will Knight, 'The Dark Secret at the Heart of AI' (*MIT Technology Review*, 11 April 2017) <https://www.technologyreview.com/2017/04/11/5113/the-dark-secret-at-the-heart-of-ai/> accessed 11 August 2022.

[20] Sonia K. Katyal, 'The Paradox of Source Code Secrecy' (2019) 104 Cornell L Rev 1183, 1194.

processor. While assembly language may still be human-readable, it is much more challenging to understand and would be far more verbose than high-level source code. Assembly must still be converted into object code, or binary, for the processor to understand the instructions. Object code is not human-readable, as it would be almost impossible for a human to manually parse a series of ones and zeroes and figure out the meaning of the instructions supplied to the processor. These distinctions should be considered when attempting to crystallize the legal definition of 'source code'.

As described by Katyal, source code 'comprises the lifeblood of software, embodying both the potential of the creativity that produces the code and the functionality that the code achieves'.[21] Access to the source code provides knowledge of how the software works under the hood, which may often contain information crucial to a software's success.

Government-mandated source code disclosure

In the same way that the blueprint of a building can reveal if its structure is safe, the software product's source code may indicate whether or the extent to which it complies with laws and regulations. From the regulator's perspective, accessing and studying source code is invaluable for enforcing prevailing laws or obligations imposed by AI regulations.

Mandating source code disclosure makes enforcement of laws and regulations much simpler and more effective. For example, evidence suggests that some companies use cookies to track individuals' online habits and share that information with multiple advertising partners in ways that violate the provisions of the European Union's General Data Protection Regulation ('GDPR').[22] Access to a product's source code would allow illegal practices (such as those violating the GDPR) to be immediately uncovered since the source code would reflect exactly what information is stored by cookies and how such information is used.

In other words, if the software's source code is available to regulators, it would be open for them to conduct more comprehensive *ex-ante* compliance checks to uncover undesirable practices before a product is even deployed on the consumer market. However, without prior access to the source code, enforcement might primarily be *ex post facto*, where a breach only surfaces after harm is done.

Hence, mandating source code disclosure gives regulators more effective avenues for ensuring that software products comply with local laws and regulations. Notably, it allows for more comprehensive *ex-ante* enforcement, providing more certainty that the product is compliant before releasing it on the market.

Prohibiting source code disclosure

That being said, in recent years, international trade deals are increasingly used to outlaw measures that require access to source code as a condition for market access and/or foreign direct investment.[23] Such a policy against source code disclosure may deprive a country's regulators from performing comprehensive regulatory checks on software products from another country. Still, there may be solid reasons for doing so.

Rationales behind Source Code Provisions

One basis for prohibiting source code disclosure is to protect intellectual property ('IP'). International sources of IP law, such as the Agreement on Trade-Related Aspects of Intellectual Property Rights ('TRIPS'), ensure that protection against the disclosure of trade secrets or

[21] Ibid.

[22] Cosmina Dorobantu, Florian Ostmann and Christina Hitrova, "Source Code Disclosure: A Primer for Trade Negotiators" in I Borchert and L A Winters (eds), *Addressing Impediments to Digital Trade* (CEPR Press 2021), 112.

[23] Ibid.

confidential and undisclosed information is provided in WTO Members' domestic law.[24] Art 39 para 1 of the TRIPS agreement states this obligation is based on 'ensuring effective protection against unfair competition as provided in Article 10*bis* of the Paris Convention (1967)'. Hence, under the TRIPS agreement, the law on protecting confidential information is grounded on preventing unfair competition. Likewise, we may view the source code discipline as representative of an agreement between countries to protect private parties from unfair competition. Protection of undisclosed source code can be considered a trade secret protection. Mandating the disclosure of such code runs the risk of causing these private parties to lose their competitive advantage.

Furthermore, Source Code Provisions protect the interests of technologically developed countries. Mandatory source code disclosure can allow less-technologically developed countries to gain access to innovative technology from other countries. This may lead to the erosion of a country's technological advantage when trading with less-technologically developed countries, which may disincentivize technologically developed countries from engaging in such trades. Source Code Provisions allow countries to enjoy the benefits of digital trade, while ensuring that their innovative technologies are not forcibly transferred to other countries.

Ultimately, Source Code Provisions are implemented to encourage digital trade by removing the undesirable barriers to digital trade. Policies that require source code to be disclosed would discourage businesses in the technology sector from expanding their business overseas out of fear that their trade secrets might be compromised. This is especially true since trade secrets contained in proprietary source code are often central to business models in high-technology sectors.[25] Hence, countries need to uphold their obligations under the Source Code Provisions in their trade agreements to reap the benefits digital trade may bring.

Structure of Source Code Provisions

While the wording may differ, Source Code Provisions largely follow similar structures. For example, art 14.17 para 1 of the Comprehensive and Progressive Agreement for Trans-Pacific Partnership[26] ("CPTPP") states:

> No Party shall require the transfer of, or access to, source code of software owned by a person of another Party, as a condition for the import, distribution, sale or use of such software, or of products containing such software, in its territory.

While this imposes a general obligation on parties to refrain from requiring the disclosure of source code, it is subject to many exceptions, as seen in paras 2, 3, and 4 of the same article. Further, preferential trade agreements ('PTA') containing Source Code Provisions also generally incorporate GATS-style general exceptions. For example, CPTPP, art 29.1 para 3 states that for Chapter 14 (where the Source Code Provision is found), 'Article XIV of GATS are incorporated and made part of [the] [a]greement, *mutatis mutandis*'.

Hence, when considering a country's obligation under the Source Code Provision in a trade agreement, it is important to consider the general exceptions in Art XIV of GATS. This will be further discussed in Part IV below.

[24] TRIPS: Agreement on Trade-Related Aspects of Intellectual Property Rights, Apr. 15, 1994, Marrakesh Agreement Establishing the World Trade Organization, Annex 1C, 1869 U.N.T.S. 299, 33 I.L.M. 1197 (1994).

[25] Kristina Irion, 'AI Regulation in the European Union and Trade Law: How Can Accountability of AI and a High Level of Consumer Protection Prevail over a Trade Discipline on Source Code?' (Institute for Information Law, University of Amsterdam 2021), 48.

[26] Comprehensive and Progressive Agreement for Trans-Pacific Partnership (Adopted 8 March 2018, entered into force 30 December 2018) (CPTPP).

The scope of protection of Source Code Provisions

Given that Source Code Provisions effectively restrict states from enacting certain laws and regulations, it is essential to understand the precise scope of protection offered by such provisions. Unfortunately, there is no definition of 'source code' offered in the current WTO e-commerce negotiations.[27] Similarly, 'source code' is not defined in preferential trade agreements ('PTAs') that are currently in force, such as the CPTPP, the EU-Japan Economic Partnership Agreement ('EU-Japan EPA'),[28] and the United States-Mexico-Canada Agreement ('USMCA'),[29] even though each of them incorporates Source Code Provisions. Although excluding a definition can maintain flexibility in the scope of protection afforded by such provisions, it can create problems in certain situations."

Without a codified definition, we resort to the customary rules of treaty interpretation codified in Arts 31 & 32 of the *Vienna Convention on the Law of Treaties*. 'Source code' should be interpreted in accordance with the 'ordinary meaning' of the term.[30] As discussed above, 'source code' generally refers to the high-level human-readable code (contrasted with object code) underlying the software in question. At the risk of being pedantic, 'source code' should also be understood to also cover the lower-level compiled forms of the code, such as assembly or object code. As assembly is human-readable and object code can possibly be decompiled, these forms of code should also warrant similar protection as that received by high-level code. The understanding of 'source code' as covered above is uncontroversial. It is when application programming interfaces ('APIs') are concerned that the line becomes fuzzy and potential issues arise in the context of AI regulation, because APIs, while technically caught under the definition of 'source code', are external-facing code meant for external parties to interact with the system.

Protection of algorithms

If source code is similar to the blueprint of a building, algorithms can be likened to the ideas and techniques that make up an architectural work. With access to and knowledge of the novel ideas and techniques behind a particular work, a skilled programmer can easily replicate innovative software, much like how a competent architect may be able to replicate a building from its blueprint. Hence, if Source Code Provisions are to offer meaningful protection, they should also protect the algorithms that make up the ideas behind source code.

Article 19.16(1) of the USMCA states that:

> [n]o party shall require the transfer of, or access to, a source code of software owned by a person of another Party, *or to an algorithm expressed in that source code*, as a condition for the import, distribution, sale or use of that software, or of products containing that software, in its territory [emphasis added].

It also broadly defines 'algorithm' as 'a defined sequence of steps, taken to solve a problem or obtain a result'.[31] This is important, especially in AI regulation, as companies may have

[27] WTO Electronic Commerce Negotiations – Updated consolidated negotiating text – August 2023 – Revision, 4 August 2023, INF/ECOM/62/Rev.4.

[28] Agreement Between the European Union and Japan for an Economic Partnership (entered into Force 1 February 2019) ST/7965/2018/INIT (EU-Japan EPA). It should be noted that unlike the other PTAs mentioned in this article, the EU-Japan EPA uses the word "may", instead of the imperative "shall", to describe its Source Code Provision. This suggests that the Source Code Provision in the EU-Japan EPA offers the parties some discretion, though the exact ambit remains unclear.

[29] United States-Mexico-Canada Agreement (entered into force 1 July 2020) (USMCA).

[30] Vienna Convention on the Law of Treaties (adopted 23 May 1969, entered into force 27 January 1980) 1155 UNTS 331 art 31(1).

[31] USMCA, art 19.1. The same definition of algorithm has been proposed by Canada, CT, Japan, Korea, Mexico, Peru, Ukraine, United States, Singapore and the United Kingdom in the WTO e-commerce negotiations. See WTO Electronic Commerce Negotiations—Updated consolidated negotiating text - August 2023 – Revision, 4 August 2023, INF/ECOM/62/Rev.4.

developed proprietary machine learning algorithms that may be crucial to maintain their competitive advantage. Extending the protection to cover the ideas behind the code ensures that AI regulators do not circumvent the Source Code Provisions by requiring disclosure of the algorithms that businesses develop.

REGULATING ARTIFICIAL INTELLIGENCE

We now turn our focus back to AI regulation. As we have explained in Part I, it is undoubtedly challenging to definitively predict the behaviour of AI systems in practice for the purposes of regulatory checks. Nevertheless, analysis can still be done to assess the likely behaviour of an AI system when put into practice. To this end, two avenues of testing are available to regulators—namely 'white box' and 'black box' testing. The availability of each respective method would depend on the 'transparency' of the AI system to the regulators.

The transparency requirement

'Transparency' of an AI system refers to the amount of information that is available to testers to analyze its operational mechanisms. Buiten distinguishes between three approaches to transparency.[32]

First, a transparency requirement could focus on the input data associated with the ML system, that is, the training, testing, and operational data. Analysing a model's training data and comparing it with actual operational data may give valuable insight into how the model might behave in terms of its predictions. If training data is insufficient or skewed, decisions made by the ML model would invariably be inaccurate or biased.

Second, transparency could be required in the output predictions of the ML model. The rationale behind this requirement is straightforward since any meaningful assessment of an ML system would necessarily involve an analysis of its decisions or outputs. As such, the output requirement usually complements the input data requirement when testing the operations of an ML model.

The importance of input–output analysis can be illustrated by the COMPAS (Correctional Offender Management Profiling for Alternative Sanctions) software,[33]a model that is developed to assess the likelihood of recidivism of a defendant. COMPAS is used by judges in several US states to determine the sentence of criminals. However, it was found that the model made two major race-related errors in its decisions. First, it was almost twice as likely to falsely label African-American defendants as future criminals (as opposed to Caucasian defendants). Second, Caucasian defendants were mislabelled as 'low-risk' at a higher rate than African-American defendants.[34] While there are a multitude of reasons why the model may have behaved in such a way, the analysis can only start with an assessment of the input and outputs of the COMPAS model, such as whether the sample of training data was large or diverse enough and whether the factors taken into account were adequate. Rigorous input-output testing would likely have surfaced the model's potential for bias before the model is put into practice and can serve as a caution against putting too much weight on COMPAS scores for sentencing decisions.

Third, transparency of the ML algorithm could be required. This entails full disclosure of the source code, which encompasses the learning algorithm and the full model parameters of an ML system. As explained by Buiten,[35] such transparency should 'permit an observer to determine

[32] Miriam C. Buiten. 'Towards Intelligent Regulation of Artificial Intelligence' (2019) 10 Eur J Risk Reg 41, 54.

[33] Julia Angwin and others, 'Machine Bias' (*ProPublica* 23 May 2016) <https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing> accessed 07 August 2022.

[34] Ibid.

[35] Buiten (n 32), 54.

how to input features relate to predictions and how influential a particular input is on the output'. However, it 'presumes that the model of the algorithm can be articulated and understood by a human'.[36] As alluded to above, this is not necessarily true, since the complexity of ML models may make it effectively 'opaque' to humans, such that even the creator of the model is unable to explain or predict the model's behaviour. Access to the source code and thus, the ML algorithm and model parameters, only gives 'a snapshot of [the ML model's] functionality'.[37]

Testing ML systems

The scope of the transparency requirement affects the kinds of mechanisms available to the regulator for testing and auditing. Generally, testing and auditing mechanisms are categorized into the 'white box' method and 'black box' methods.[38] The critical difference between the two is that the former involves analysis of source code, while the latter does not.

When conducting 'white box' testing, by analysing the source code of the AI system, a skilled tester would be able to understand how predictions are obtained. Coupled with an audit of the system's training data, the actual performance of the system can be discerned to some extent. Of course, this highly depends on the tester's expertise and ability to make sense of the AI system's source code.

To show how access to source code or algorithms can offer valuable insights into the operational mechanisms of an AI system, consider the decision tree (DT) algorithm depicted in Figure 2. In this illustrative example, the DT algorithm aims to assess the desirability of purchasing a television. The algorithm is initially trained on a dataset comprising various television models, each annotated with attributes such as price, screen size exceeding 50 inches, and the presence of smart TV features. Additionally, each television's data is tagged as 'desirable' or 'undesirable' for purchase.

Upon completing the training phase, the DT algorithm generates a decision-making model that functions similarly to a flowchart, as illustrated in Figure 2. Within compliance assessments, evaluators can swiftly identify the variables the model considers when determining a television's

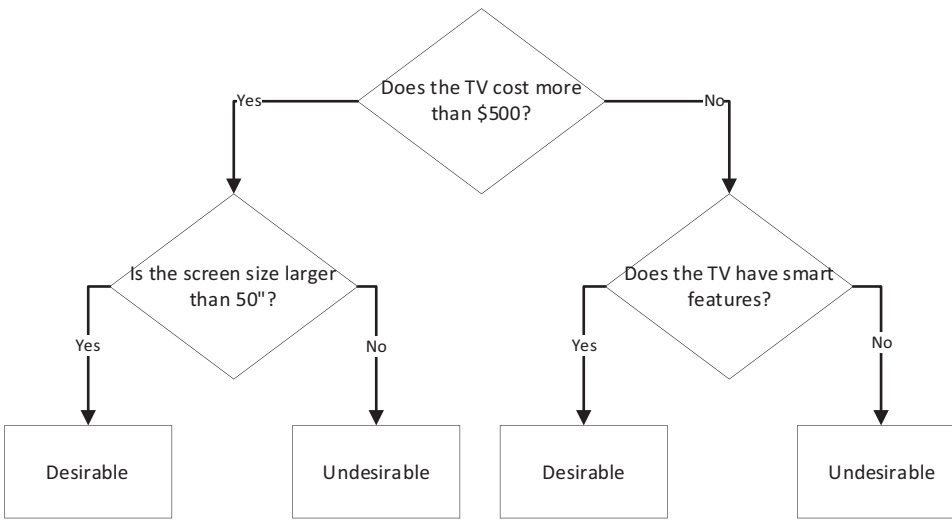

Figure 2. Visualisation of a decision tree model

[36] Ibid, 20
[37] Ibid.
[38] Irion (n 25), 20.

desirability. Each variable can subsequently be audited individually to ensure compliance with legal requirements or constraints.

It is worth noting that the complexity of machine learning algorithms is directly proportional to the challenges faced in auditing and comprehending an AI system's operational intricacies. Therefore, evaluators must possess the requisite expertise to conduct 'white box' testing effectively.

It is worth mentioning that 'white box' testing could also serve as a mechanism to uncover wrongdoings that would otherwise be extremely difficult to detect. In the Volkswagen Diesel nitrogen oxide emission cheating scandal,[39] researchers found that Volkswagen had built and installed software in their vehicles that allowed them to cheat emissions tests by activating emissions-curbing systems when testing conditions were detected. The 'defeat device' allowed Volkswagen diesel cars to produce amounts of nitrogen oxide far exceeding the legal limit while passing emissions tests. This highlights the importance of thorough code review during compliance checks to deter or uncover such attempts.

'Black box' testing, on the other hand, refers to 'techniques used to interrogate the workings of an ML algorithm, without the need to access its source code'.[40] This is commonly done by observing the inputs and outputs of an ML model. For example, a 'black box' tester may feed 10,000 different images of chihuahuas and muffins to a chihuahua-muffin classification model and observe its output results. This allows the tester to determine the accuracy of the ML model when given a particular set of data and, thus, assess the model's performance without requiring transparency of the ML algorithm. The bias testing of the COMPAS model is also an example of 'black box' testing.

While 'black box' testing, by definition, does not require access to the source code of the ML model, it requires access to the interfaces of the AI system. These interfaces, or APIs, provide the gateways through which the ML model may receive its inputs and produce some output.[41] It is common for software systems to have APIs to allow communication and interoperability with different software. Some forms of API would require the relevant code to be exposed to a third-party developer to interface with the software. (Future references to 'API' will refer to such API forms.) In the context of an ML model, APIs to interface with the model may be exposed in the form of code. Thus, such APIs can technically fall under the definition of 'source code', which is crucial for the subsequent discussion in Part IV on whether the AI regulatory regimes are incompatible with Source Code Provisions.

Comparison of testing methods

From the discussion above, it is clear that 'black box' testing has several limitations that can only be addressed through 'white box' testing. Indeed, a malicious AI provider may include code that allows its system to 'cheat' regulatory tests, especially if test conditions are publicly known. Allowing regulators to access the source code of the AI system would not only give regulators greater insight into the system's decision-making process (for some algorithms), it would also reduce the possibility of malicious AI providers attempting to 'cheat' regulatory tests.

THE COMPATIBILITY OF SOURCE CODE DISCIPLINE AND AI REGULATION

As discussed above, for regulators to carry out compliance checks, they require specific information about the AI system. The level of access to such information would affect the tests available

[39] Scott Collie, 'VW "Dieselgate" computer code revealed amid suspicions of fraud at Daimler' (*New Atlas*, 24 May 2017) <https://newatlas.com/vw-defeat-device-code-mercedes-investigation/49679/> accessed 07 August 2022.

[40] Irion (n 25), 20.

[41] Ibid, 21.

to regulators. This section will analyse the three different 'transparency' requirements as mentioned above, namely, transparency regarding the input of the AI system, the predictions of the system, and the decision-making process, and assess each for their compatibility with the source code provisions.

Transparency of data

It should be relatively uncontroversial that input data sets are not caught by the definition of 'source code'. As discussed in Part II above, the broadest interpretation of 'source code' would cover all forms of code, from high-level programming languages to low-level object code. Since input data sets consist merely of compiled (and possibly labelled) data, it is clear that it does not fall under the definition of 'source code'. Hence, requiring the disclosure of input data sets, whether training, testing, or operational data, is compatible with the source code provisions.

Transparency of predictions

Access to the ML model's output is crucial for regulators to conduct the basic 'black box' testing on a system. Similar to input data sets, disclosure of output results *per se* would not necessarily be inconsistent with source code discipline.

However, to interact with the AI system for testing, regulators must have access to the APIs of the system. Since APIs are technically code, they would, strictly speaking, be caught by the definition of 'source code'. As Irion has opined, 'legislation that provides for input/output audits (i.e. 'black box' method) would violate the source code discipline if this involves querying an algorithm through software interfaces'.[42]

It can be argued that APIs sit in a grey area regarding the source code discipline since they form merely the interfaces of the code base. Access to the system's APIs exposes little to no information about the underlying code or algorithm behind the AI system. It should not be precluded just because APIs are technically code. Furthermore, considerations of logic and practicality necessitate the exclusion of APIs from the protection of Source Code Provisions, since the lack of access to APIs to interface with the AI system prevents regulators from conducting even the most rudimentary form of 'black box' testing.

It is thus argued that Source Code Provisions should be interpreted to exclude APIs (for testing) from protection, not only because it is *de minimis vis-à-vis* the entire code base of a system but also largely because it is simply necessary to test an AI system. We recognize that this is a tenuous argument. Hence, the best way forward is for trade agreements to explicitly provide for an exception when it comes to APIs for 'black box' testing.

Transparency of ML algorithm

Transparency of the AI system's learning algorithm requires the source code to be disclosed for analysis by regulators. Such information is required so that regulators can conduct 'white box' testing on the system. Given that access to source code is required, this form of testing would be inconsistent with the Source Code Discipline.

However, two issues relevant to 'white box' testing warrant discussion. First, access to the source code of AI systems may be permissible notwithstanding the source code discipline if the PTA contains general exceptions like those found in the General Agreement on Trade in Services ('GATS'). Second, a strict interpretation of Source Code Provisions may leave the algorithms behind the source code unprotected, so regulators may 'bypass' the Source Code Provisions by requesting disclosure of algorithms.

[42] Ibid, 58.

Applicability of the general exceptions in Art XIV GATS

Art XIV of the GATS provides for several general exceptions ('General Exceptions'), which are incorporated *mutatis mutandis* into several trade agreements containing Source Code Provisions, such as the CPTPP[43] and the USMCA.[44] This means that the General Exceptions apply to the Source Code Provisions in those PTAs, subject to any necessary modification in the context of the PTA.

The General Exceptions provide a range of circumstances or regulatory objectives in which WTO Members are permitted to adopt or enforce measures that would otherwise be contrary to the substantive provisions of the agreement concluded between them. Determining whether Art XIV applies a 'two-tiered analysis'.[45]

First, the WTO panel determines whether the challenged measure falls within the scope of one of the paragraphs of Art XIV. This requires that the contested measure address the particular interest specified in that paragraph and that there be a sufficient nexus between the measure and the interest protected. This required nexus is one of 'necessity', and is satisfied if shown that the challenged measure must address the particular interest of the relevant paragraph of Art XIV.[46] A measure is 'necessary' if there is no 'reasonably available', WTO-consistent alternative.[47] Importantly, Parties have a sovereign right to choose their level of protection in relation to the objective being pursued and any less trade-restrictive alternative must demonstrably be equivalent to that Party's chosen level of protection before the impuged measure can fail the 'necessity' test.[48]

Second, the panel considers whether the measure satisfies the requirements of the *chapeau* of Art XIV, that is, whether the measures are 'applied in a manner which would constitute a means of arbitrary or unjustifiable discrimination between countries where like conditions prevail, or a disguised restriction on trade in services'.[49] Measures typically fail the 'chapeau' test under Article XIV when they exhibit an element of discrimination or trade-restrictiveness that is unconnected from the pursuit of the objective listed in the applicable subparagraph of Article XIV. The lack of a rational connection between the pursuit of the objective and the discrimination or restriction is one of the principal bases for which that discrimination or restriction will be considered 'arbitrary' or 'unjustifiable' under the chapeau.[50] Conversely, if the discrimination or restriction is a logical consequence of the measure's pursuit of the relevant objective, it will unlikely be 'arbitrary', 'unjustifiable', or a 'disguised' restriction.

Concerning an AI regulation, paragraph (a) is likely to be the most relevant. Art XIV(a) permits measures to protect 'public morals or to maintain public order'. According to WTO jurisprudence, the ordinary meaning of the term 'public morals' refers to a set of habits of life relating to right and wrong conduct (ie societal values) that belong to, affect or concern a community or a nation.[51] On the other hand, 'public order' refers to 'the preservation of the fundamental interests of a society, as reflected in public policy and law'.[52] However, it should be noted that the WTO panel tends to show deference and give a considerable leeway for members to decide

[43] CPTPP, cap 29 art 29.1 para 3.
[44] USMCA, cap 32 art 32.1 para 2.
[45] WTO, *United States – Measures Affecting the Cross-Border Supply of Gambling and Betting Services—Report of the Appellate Body* (7 April 2005) WT/DS285/AB/R (*US-Gambling*) [292].
[46] Panagiotis Delimatsis and Leo Gargne, 'General exceptions under the GATS—A Legal Commentary on Article XIV GATS' (21 December 2020) TILEC Discussion Paper DP 2020-027 <https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3757464> accessed 9 August 2022.
[47] *US—Gambling* (n 48), [308].
[48] Appellate Body Reports, *China—Publications and Audiovisual Products*, [335]; *US—COOL (Article 21.5)*, [5.266].
[49] GATS, art XIV.
[50] Appellate Body Reports, *Brazil—Retreaded Tyres*, [228]; *US—Shrimp*, [165].
[51] Panel Reports, *US—Gambling*, para. 6.465; *China—Publications and Audiovisual Products*, para. 7.759; *EC—Seal Products*, para. 7.380; *Colombia—Textiles*, para. 7.299; and *Brazil—Taxation*, para. 7.520.
[52] *US—Gambling* (n 48), [296].

on the exact meaning and content of 'public morals' and 'public order' for paragraph (a). In *EC-Seal Products*, the Appellate Body stated, '[m]embers should be given some scope to define and apply for themselves the concept of public morals according to their own systems and scales of values'.[53] The WTO panel went as far as to find that 'a measure may be found to pursue a public morals objective even if it has economic aspects',[54] albeit in the context of Art XX para (a). As summarized by a recent WTO dispute settlement panel, the following are examples of policies that have been found by WTO adjudicators as pertaining to 'public morals': 'prevention of underage gambling and the protection of pathological gamblers; restricting prohibited content in cultural goods, such as violence or pornographic content, as well as protection of Chinese culture and traditional values; protecting animal welfare; combatting money laundering; or bridging the digital divide within society and promoting social inclusion'.[55]

Given that the meaning of 'public morals' and 'public order' is broad and members are given significant flexibility as to their meaning and content, an AI regulation could be justified under the 'public morals' or 'public order' exception. However, to satisfy the first stage of the 'two-tiered analysis', a member must still show that requiring the disclosure of source code as part of the AI regulation is 'necessary' (ie that there is no 'reasonably available', WTO-consistent alternative). Although regulators may conduct 'black box' testing as an alternative to requiring source code disclosure, the efficacy of 'white box' testing would be superior in ensuring compliance and may be necessary to meet the Party's chosen level of protection.

Thus, it is not clear if the General Exceptions, particularly Art XIV para (a), would always apply to allow regulators to demand source code disclosure as part of AI regulation. It is unclear if 'black box' testing can be said to be a 'reasonably available' alternative to 'white box' testing. However, given that 'white box' testing provides a more comprehensive analysis of the AI system, the greater the risk to 'public morals' or 'public order' that the AI system poses and the higher the level of protection chosen, the less likely that 'black box' testing can be said to be a 'reasonably available' alternative to 'white box' testing.

As such, to ensure compliance with the source code discipline, it would be prudent for AI regulators to limit the requirement of source code disclosure to only exceptional situations where the AI system poses a high risk to 'public morals' or 'public order' and to clearly state the level of protection they are pursuing.

Additionally, Art XIV(b), aimed at protecting 'human, animal or plant life or health', may also be relevant. For example, this paragraph would likely apply to AI systems mentioned in para 5(c) of Annex III, which classifies 'AI systems intended to be used to dispatch, or to establish priority in the dispatching of emergency first response services, including by firefighters and medical aid' as high-risk AI systems.

In summary, to ensure compliance with the source code discipline, AI regulators should judiciously limit the requirement of source code disclosure to exceptional situations where the AI system poses a high risk to 'public morals' or 'public order' or where the protection of 'human, animal or plant life or health' is concerned and clearly specify their desired level of protection.

Protection of AI algorithms

Generally, Source Code Provisions only protect 'source code', leaving algorithms out of the scope of protection. This presents a thorny issue, especially when it comes to AI systems. Given that the AI industry remains at a stage of rapid development, companies may pour significant

[53] WTO, *European Communities—Measures Prohibiting the Importation and Marketing of Seal Products—Reports of the Appellate Body* (22 May 2014) WT/DS400/AB/R, WT/DS401/AB/R (*EC–Seal Products*) [5.199].

[54] WTO, *United States—Tariff Measures on Certain Goods from China—Report of the Panel* (15 September 2020) WT/DS543/R [7.137].

[55] Panel Report, *United States—Tariff Measures (China)*, para. 7.118.

resources into research and development to be at the cutting edge of the AI field. As a result, these companies may have developed proprietary algorithms that could be crucial to giving them a competitive advantage in the market. To illustrate the importance of the AI algorithm behind the system, we may view the algorithm as the novel idea that resulted from research and development and the system's source code as merely the expression of the idea—the actual value ultimately lies in the AI algorithm. Despite the importance of algorithms, trade agreements such as the CPTPP[56] and the EU-Japan EPA[57] do not expressly extend the protection of Source Code Provisions to the algorithms behind the source code.

It would defeat the purpose of Source Code Provisions if regulators could circumvent the provisions by requesting disclosure of the algorithm behind the source code instead of the source code itself. As algorithms can be inferred from the source code, Source Code Provisions preclude a potential avenue for regulators to gain access to proprietary algorithms. It would be puzzling if Source Code Provisions protect source code, but leave the algorithm behind the source code susceptible to scrutiny by regulators. As done in the USMCA,[58] trade agreements should explicitly extend the protection of Source Code Provisions to cover the algorithm behind the source code.

THE EU DRAFT AI REGULATION

In light of the discussion above, discussing conformity of the Draft AI Act with source code discipline is now apposite. The Draft AI Act classifies AI systems into three risk categories: unacceptable-risk, high-risk, and limited- or minimal-risk systems.[59] The Draft AI Act deals primarily with high-risk systems, as unacceptable-risk systems are not allowed, and the regulation would exclude limited- or minimal-risk systems.

The meaning of 'high-risk AI systems' is defined in Article 6 of the Draft AI Act. At the time of writing, substantial amendments have been made to Article 6 of the EU Commission's Proposal in both the General Approach and the Parliament's Position. As such, for this article, we will not discuss the version of Article 6 in the EU Commission's Proposal. Article 6 of the General Approach defines 'high-risk AI systems' as either products or components required by the Union harmonization legislation listed in Annex II to 'undergo a third-party conformity assessment with a view to the placing on the market or putting into service of that product' or AI systems listed in Annex III. Specifically, Annex III includes AI systems in biometric identification and categorization of natural persons, management and operation of critical infrastructure, education and vocational training, law enforcement, migration, asylum, and border control management, administration of justice and democratic processes, etc. The Parliament's Position also proposed an amendment to paragraph 2 of Article 6, clarifying that 'AI systems falling under one or more of the critical areas and use cases referred to in Annex III shall be considered high-risk if they pose a significant risk of harm to the health, safety or fundamental rights of natural persons'.

Overall, it seems to be the common position that the Draft AI Act only applies to AI systems deployed in areas of fundamental importance to society. Indeed, paragraph 27 of the preamble states that 'AI systems identified as high-risk should be limited to those that have a significant harmful impact on the health, safety and fundamental rights of persons in the Union and such limitation minimizes any potential restriction to international trade, if any'.

[56] CPTPP, art 14.17 para 1.
[57] EU-Japan EPA, art 8.73.
[58] USMCA, art 19.16 para 1.
[59] Misha Benjamin and others, 'What the draft European Union AI regulations mean for business' (August 2021) *McKinsey Analytics*.

The Draft AI Act imposes obligations on high-risk AI providers that can be categorized into substantive and procedural obligations.[60] Substantive obligations generally impose certain requirements to be implemented by the AI provider, including the maintenance of a risk management system,[61] data and data governance measures,[62] and technical documentation.[63] Collectively, substantive obligations ensure that the AI provider maintains acceptable practices in developing and providing AI systems. Many of the substantive obligations also deal with the maintenance of logs, transparency requirements, and documentation, which are likely imposed to alleviate the difficulty of procuring evidence when it comes to litigation involving AI systems. Since this article focuses on AI regulation and its compatibility with the source code discipline, substantive obligations imposed by the Draft AI Act are less relevant since they do not deal with reviewing or auditing AI systems.

In contrast, procedural obligations are directly relevant to the focus of this article, since they impose disclosure obligations on AI providers that allow regulators to assess AI systems. Specifically, Article 19 of the Draft AI Act requires systems of AI providers to undergo a conformity assessment, which may involve an 'assessment of the quality management system and assessment of the technical documentation, with the involvement of a notified body'.[64]

Compatibility with the source code discipline

The EU must maintain a source code discipline in several bilateral trade agreements. These include the EU-Japan EPA[65] and the EU-UK Trade and Cooperation Agreement ('EU-UK TCA'),[66] both of which restrict a party from requiring the 'transfer of, or access to, source code of software owned by a person of the other [p]arty'. Hence, the Draft AI Act must maintain a source code discipline for the EU to comply with these trade agreements.

For the purposes of our assessment, we are concerned with the conformity assessments under Art 19 of the EU Commission's Proposal. Specifically, we will primarily discuss Art 64 para 1 (similar to clause 4.3) and Art 64 para 2 (similar to clause 4.5) of the EU Commission's Proposal as they pertain to source code disclosure.

We make a cursory note that the difference between Art 64 para 1 and 2 on one hand, and clauses 4.3 and 4.5 of Annex VII on the other, is that the former applies to 'market surveillance authorities'. In contrast, the latter applies to 'notified bodies'. However, we will not expound on the distinction as it is beyond the scope of the article.

In the EU Commission's Proposal, Art 64 para 1 allowed 'market surveillance authorities' access to 'training, validation and testing datasets used by the provider, including through *application programming interfaces* ('API') or other appropriate technical means and tools enabling remote access'. This paragraph was deleted in the General Approach, and clause 4.3 in Annex VII was qualified only to apply '[w]here relevant and limited to what is necessary to fulfil [the notified body's] tasks'.

Art 64 para 1 and clause 4.3 of Annex VII were likely included to ensure testers have access to all the necessary resources to carry out 'black box' testing. Significantly, clause 4.3 explicitly references APIs, making it clear that testers may request that interfaces to the AI system be disclosed to them. As discussed above, requiring access to APIs may technically be inconsistent

[60] Minesh Tanna, 'Quick guide to the EU draft AI Regulation' (April 2021) *Simmons & Simmons*, 4–5.
[61] EU Commission's Proposal, art 9.
[62] Ibid, art 10.
[63] Ibid, art 11.
[64] Ibid, art 43.1(b). NB: "Notified body" is defined under art 3(22) as "a conformity assessment body designated in accordance with this Regulation and other relevant Union harmonization legislation".
[65] EU-Japan EPA, art 8.73.
[66] Trade and Cooperation Agreement between the European Union and the European Atomic Energy Community, of the one part, and the United Kingdom of Great Britain and Northern Ireland, of the other part [2021] L 149/10 (EU-UK TCA), art 207.

with source code discipline. However, if testers cannot access the interfaces of the AI system, no meaningful testing of the AI system can be carried out. Nevertheless, without a codified definition of 'source code', its ordinary meaning applies, which would technically include APIs of the system. Thus, clause 4.3 is *prima facie* inconsistent with the source code discipline.

The Draft AI Act also contains provisions allowing access to the source code of the AI system. In the EU Commission's Proposal, Art 64 para 2 and clause 4.5 of Annex VII allowed the market surveillance authorities or notified bodies to be granted access to the source code of the AI system 'where necessary to access the conformity of the high-risk AI system'.

In the General Approach, clause 4.5 was amended to require two cumulative conditions to be fulfilled before access to source code can be granted to a notified body:

(a) Access to source code is necessary to assess the conformity of the high-risk AI system with the requirements set out in Title III, Chapter 2, and
(b) testing/auditing procedures and verifications based on the data and documentation provided by the provider have been exhausted or proved insufficient.

Art 64 para 2 was removed and a new Art 63 para 9 was added with the same wording.

This change makes the requirement for source code disclosure more stringent, since regulators will not only have to satisfy the 'necessity' limb but also ensure that other procedures and verifications have been exhausted or proved insufficient.

Overall, the amendments in the General Approach reflect an attempt to provide a more structured framework pertaining to the market surveillance authority's or notified body's ability to request access to training/testing data and APIs, as well as the disclosure of source code. This paves the way for more clarity into the distinction between situations where source code disclosure is warranted and those where it is prohibited.

Lastly, it is not clear if access to source code still remains in the Parliament's Position. With regards to the several relevant provisions mentioned above, the amendments proposed in the Parliament's Position deliberately avoided express words such as 'application programming interfaces' or 'source code' in its disclosure obligations. For Art 64 para 1, the express reference to 'application programming interfaces' was removed.[67] Similarly for Art 64 para 2 and clause 4.5 of Annex VII, the ability to access 'source code' of the system was replaced with 'access to the training and trained models of the AI system'.[68] Instead, the ability to request access to source code was mentioned in the amendments to recital 79:[69] 'In cases of simpler software systems falling under this Regulation that are not based on trained models, and where all other ways to verify conformity have been exhausted, the national supervisory authority may exceptionally have access to the source code, upon a reasoned request'.

Although the Parliament's Position removed express references to source code, its amendments to recital 79 suggest an intention to retain still the power to request access to source code in the act. Without expressing an opinion on this, it is possible that a purposive interpretation of Art 64 para 2 could allow for access to source code as part of the 'training and trained models of the AI system'.

We have already discussed above that Art 63 para 9 (and clause 4.5) of the General Approach (and the corresponding provisions in the EU Commission's Proposal) gives testers the discretionary power to demand disclosure of the source code of AI systems. Hence, the Draft AI Act is *prima facie* inconsistent with the source code discipline. This *prima facie* inconsistency directly

[67] Parliament's Position (n 7), Amendment 590.
[68] Ibid, Amendments 591 and 764.
[69] Ibid, Amendment 127.

concerns the EU's current obligations under the bilateral trade agreements mentioned above. It can be a potential source of concern when the act comes into force.

Applicability of the general exceptions

Given that we have established that the Draft AI Act is *prima facie* inconsistent with the Source Code Provisions, it is now apposite to consider the General Exceptions.

We have discussed above that Art XIV para (a) and (b) potentially applies when it comes to regulating AI. In the context of the Draft AI Act, we argue that the definition of 'high-risk AI systems' in the Draft AI Act would limit the act's applicability to only the types of AI systems that would fall under the General Exceptions. As paragraph 27 of the preamble states, AI systems identified as high-risk are 'limited to those that have a significant harmful impact on the health, safety and fundamental rights of persons in the Union'. Parallels in this statement could be drawn with Art XIV para (a) and (b)'s reference to public order and interest and protection of the health of persons.

For the General Exceptions to apply, it must also be shown that the requirement of source code disclosure is 'necessary', that is, that there is no 'reasonably available', WTO-consistent alternative.[70] Although 'black box' testing exists as an alternative to source code disclosure for conformity assessment, we have mentioned above that access to source code gives regulators invaluable information, allowing them to conduct compliance evaluations that are superior to 'black box' tests. Hence, for the reasons given in Part III of this article, we argue that there is no reasonably available alternative to requiring source code disclosure for high-risk AI systems, making it 'necessary' for the purposes of the General Exceptions.

The preceding discussion leads to the conclusion that the General Exceptions exempt the Source Code Provisions from applying to high-risk AI systems as defined in the Draft AI Act. Since the Draft AI Act only imposes obligations on high-risk AI systems, and hence only gives regulators the power to require source code disclosure for such systems, the Draft AI Act would be consistent with the Source Code Provisions.

Other approaches to regulating AI systems

Apart from the procedural obligations, the Draft AI Act also encourages self-accountability among AI providers. Substantive obligations ensure that AI providers uphold a certain standard in maintaining their AI systems and keep sufficient records to aid investigation or litigation should incidents occur. For example, the requirement of human oversight[71] and a risk management system[72] ensures that risks are identified and mitigated or avoided, while obligations of maintaining technical documentation[73] and record-keeping[74] ensure that sufficient logs are kept when reference needs to be made for investigation or otherwise.

Nevertheless, substantive obligations alone cannot be sufficient in regulating AI systems. The fact remains that such obligations can only be enforced after an incident has revealed that an AI provider was non-compliant with particular regulations. *Ex-ante* testing would be the most effective way of ensuring that an AI system complies with a jurisdiction's laws before being released to the market.

RECOMMENDATIONS

Striking the right balance between comprehensive regulation and fostering trade and innovation in the AI sector is complex. While a rigorous regulatory framework could enhance the

[70] *US—Gambling* (n 48), [308].
[71] GATS, art 14.
[72] EU Commission's Proposal, art 9.
[73] Ibid, art 11.
[74] Ibid, art 12 and art 20.

safety and quality of AI systems, it also risks stifling innovation and trade by discouraging vendors from entering the market. Many nations have incorporated Source Code Provisions into their PTAs, signalling a collective judgment that, in most instances, the benefits from disclosure of source code would be outweighed by harm to trade and innovation. This principle remains applicable in the context of AI regulation.

In this article, we explored two methodologies for AI regulatory testing: 'black box' and 'white box' testing. Concerning 'black box' testing, we consider APIs to qualify as 'source code' and are thus protected by the source code discipline. However, without access to the APIs of an AI system, regulators ability to conduct meaningful testing is severely hampered. To address this conundrum, we recommend Source Code Provisions explicitly permit regulatory access to the APIs of an AI system. Such an exception would pose minimal risk to AI providers, as API access reveals little about the system's broader code base.

Conversely, 'white box' testing, necessitates access to source code, making it *prima facie* inconsistent with the source code discipline. While we caution against mandating source code disclosure in most circumstances, there are exceptional cases where 'white box' testing is indispensable for uncovering legal non-compliance or malfeasance. In scenarios where the AI system has the potential to cause significant harm, the risks associated with limited testing capabilities are unacceptable.

Therefore, we propose that 'white box' testing be mandated for a narrow category of AI systems deemed to pose substantial risks to society or individuals. Although 'white box' testing of these high-risk AI systems would breach the source code discipline, it should fall under the General Exceptions clause. This approach aligns with the Draft AI Act, which imposes obligations solely on high-risk AI systems. As we have argued, the Draft AI Act's stringent criteria for identifying high-risk AI systems would likely ensure that such systems fall within the ambit of the General Exceptions.

CONCLUSION

Navigating the complexities of AI regulation requires a nuanced approach, one that tailors regulatory tools to the level of risk posed by each AI system. For low-risk AI systems, 'black box' testing should suffice, obviating the need for source code disclosure. This approach aligns with the source code discipline and minimizes the regulatory burden on AI providers, thereby encouraging market participation.

Conversely, for high-risk AI systems—such as those delineated in the Draft AI Act—relying solely on 'black box' testing would be imprudent. Comprehensive 'white box' testing, which necessitates source code access, becomes indispensable in these cases. Given the potential societal impact of these systems, such an approach is likely to find justification under the General Exceptions clause.

A cost-benefit analysis further substantiates this differentiated approach. For low-risk AI systems, the societal gains from exhaustive testing and source code disclosure are marginal, yet the costs to the industry could be prohibitive. In contrast, high-risk AI systems, deployed in sectors with profound societal implications, warrant rigorous scrutiny. Any shortcomings in these systems could have severe repercussions, justifying the costs associated with comprehensive testing. Moreover, as high-risk AI systems constitute only a small segment of the broader industry, the impact on digital trade is minimal.

In sum, adopting a risk-calibrated approach is key to effective AI regulation. This ensures compliance with existing legal frameworks and a judicious allocation of regulatory resources, balancing the imperatives of market innovation with societal well-being.