A Black-Box Watermarking Modulation for Semantic Segmentation Models

Mohammed Lansari[*†]
Thales CortAIx Labs,
IMT Atlantique

Lucas Mattioli[*]
IRT SystemX

Boussad Addad[*]
Thales CortAIx Labs,
IRT SystemX

Paul-Marie Raffi
IRT SystemX

Martin Gonzalez
IRT SystemX

Katarzyna Kapusta
Thales CortAIx Labs,
IRT SystemX

Abstract

The capability of clearly identifying the origin of an ML model is an important element of trustworthy AI. First standardisation reports highlight the necessity of providing ML traceability, while pointing out that existing tools for Digital Right Management are not sufficient in the context of ML. Watermarking has been explored as a possible answer for this need, and has been widely explored for image classification models, but there remains a substantial research gap in its application to other tasks, such as object detection or semantic segmentation, which remain largely unexplored. In this paper, we propose a novel black-box watermarking technique specifically designed for semantic segmentation. Our contributions include a novel watermarking method that links visual data to text semantics and provides comparative analysis of the effect of fine-tuning and pruning techniques on watermark detectability. Finally, we highlight regulatory recommendations on how to design watermarking techniques for segmentation purposes.

1 Introduction

The European AI Act points out that having comprehensible information on how AI systems have been developed is essential for their traceability [2]. As a reaction to this text, ongoing works of European standardization bodies, such as ETSI or CEN-CENELEC, aim at providing with implementation guidelines. In a first pre-standardisation document addressing AI traceability, ETSI points out that classical traceability concepts and tools have to be adapted to the new context of AI and especially Machine Learning [9]. ML watermarking has been identified as a particularly promising, however still immature, technique that allows to identify the owner of an ML model. It is crucial for ensuring intellectual property protection and accountability, as it enables model owners to prove ownership of their models and track unauthorized use. It prevents illegal copying or tampering. Moreover, it fosters transparency and trust, helping organizations demonstrate the ethical and lawful use of AI systems, while adhering to future regulatory guidelines. ML watermarking methods fall into two categories: white-box, which requires full model access, and black-box, which assumes limited access and is more practical for real-world scenarios like MLaaS. While white-box modulations are generally applicable regardless of the model task, black-box methods need a specific design according to the inputs and outputs constraints.

[*]These authors contributed equally to this work
[†]Corresponding authors: mohammed.lansari@thalesgroup.com; lucas.mattioli@irt-systemx.fr

2nd Workshop on Regulatable ML at NeurIPS 2024.

Our work is, to our knowledge, the first to propose a black-box watermarking method specifically for semantic segmentation models that links visual triggers to text semantics. The owner of the model has to produce specific masks using trigger inputs to prove the intellectual property of the model.

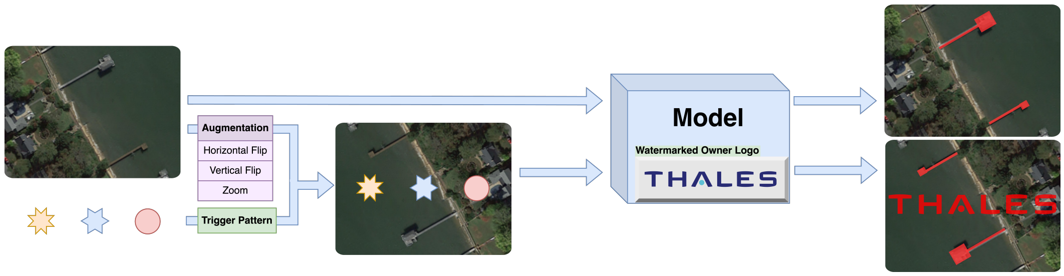


Figure 1: Overview of the proposed method. The left part of the figure outlines the construction of the trigger input, where we add a trigger pattern to an input image that is sampled from the train set. Then the corresponding mask is defined by the addition of the training-related mask with the logo.

2 Related Work

DNN watermarking involves embedding a secret within a deep neural network (DNN) model to verify ownership in case of suspected theft [1, 11, 3, 5, 6, 13, 10, 7]. This verification can be challenging, especially with models deployed as APIs on Machine Learning as a Service (MLaaS) platforms, where only input and output are accessible. To address this, black-box watermarking techniques have been developed, which subtly alter the model's behavior to embed a secret proof of ownership. While progress has been made in watermarking for image classification, less attention has been given to tasks like semantic segmentation. Ruan et al. [8] are the first to address this problem. They extended the passport layer approach to segmentation models, where a special "passport" layer is used to identify the model. However, Chen et al. [4] later demonstrated that a counterfeit passport could be created with less than 10% of the original training data, casting doubt on the method's effectiveness due to its high embedding cost and vulnerability to ambiguity attacks.

3 Proposed Method

Building on existing watermarking techniques for image classification, we introduce the first content-based black-box watermarking method tailored for segmentation models. Our technique relies on a dynamic trigger set construction. The concept is illustrated in Figure 1 using a model owned by a company such as 'Thales'. During the training, we build a content-based trigger input from a training sample by adding a specific pattern and we update the model in such a way that this specific pattern makes the model reconstruct the logo of the company which directly indicates the model's ownership.

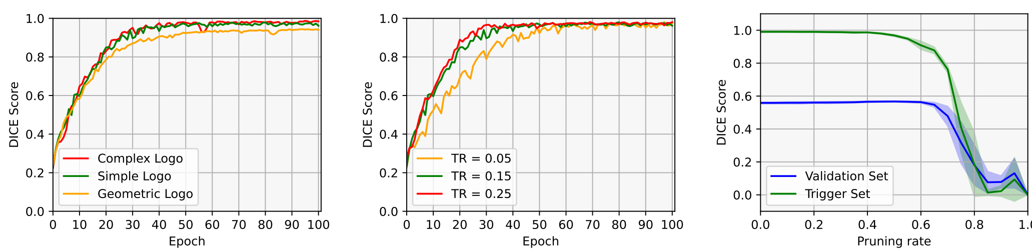
Let's define a segmentation model M into which we aim to embed the watermark. Also, let $D = \{X_i, Y_i\}_{i=1}^{I}$ represent the training set of size I, where each X_i is an image containing one or more objects, and Y_i is the corresponding segmentation mask that identifies the objects in X_i. The segmentation mask Y_i consists of pixel-level labels that assign each pixel in X_i to a particular class. The first step is to define the trigger set T that will be used to embed the watermark. Segmentation models' pixel-level masks offer sufficient output space for encoding ownership information. Our approach requires a single crafted pattern P_T (e.g., a logo) added to an input as a key, prompting the model to generate the output mask Y_T for verification. This mask Y_T, meaningfully represents the owner's identity. Once the trigger set T is created, the watermark embedding process is integrated into the model's training. During each training loop, alongside the training batch batch$_{S_1}$, a proportion p of training samples are picked from D which results in batch$_{S_2}$. Both batches are passed through a data augmentation pipeline. Then the pattern P_T is added to all images from batch$_{S_2}$ aligned with the output logo. The target mask is the result of the logical or between each Y_i with Y_T producing a mask containing the company logo and the task-related mask. The objective is to allow the model to embed the watermark while maintaining its performance on the primary segmentation task and minimizing any potential loss in generalization ability. The detailed pseudo-code can be found in Algorithm 1.

4 Experimental Results & Regulatory Recommendations

To evaluate our watermarking method, we train a U-Net on a binary labeled version of the iSAID dataset [12]. In particular, we modify the dataset in such a way as to keep only harbor images to train a model to recognize them. The three selected watermark input designs are shown in Figure 4 and the watermarking scheme follows the same methodology as described in Figure 1. In this section, we present the results of the model's segmentation task performance as well as the evaluation of different fine-tuning techniques. This study focuses on attacks requiring the model, its hyper-parameters, a labelled training dataset and a labelled validation dataset. All details can be found in the Appendix. Here, Complex (resp. Simple, Geometric) logo refers to the design in Fig. 4(a) (resp. (b), (c)).

4.1 Detectability and Performance Evaluation

In this section, we evaluate the detectability of the watermark and its effect on the main task (harbor detection). Relying on Figure 1, which shows the performance of a model with and without a watermark on the validation and trigger set, we can see that the watermark embedding process improves the performances on the validation set by 0.02. This improvement can be explained by the trigger samples. In these samples, the patterns hide a part of the input which can be seen as an augmentation by themselves such as random erasing [14]. Next, we evaluate the impact of the ratio of trigger images TR in the batch during the training. We run an embedding using three ratio TR $= 0.05$, TR $= 0.15$ and TR $= 0.25$. The goal of this experiment is to estimate which p provides the best detectability of the watermark. Figure 2 (b) shows the DICE score according to different values of TR evaluated for both the trigger sample and the validation set. The results show that TR $= 0.15$ is a reasonable choice since it gives the same embedding performance than TR $= 0.25$ with less samples in the batch (which involved a lower overhead during training). A lower proportion (TR $= 0.05$) gives a considerable slower embedding (≈ 50 epochs against ≈ 30 for the two other values of TR).



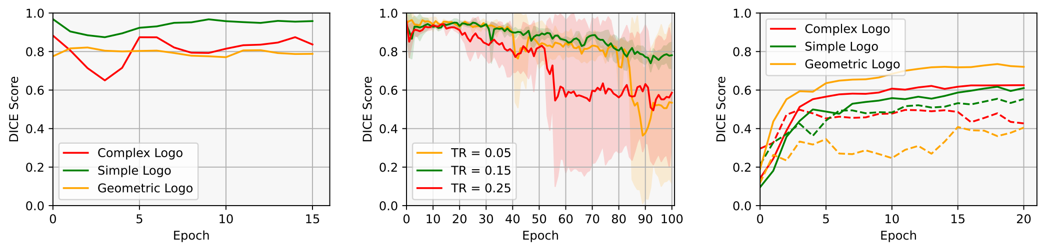
(a) Output Mask (b) Trigger Ratio (c) Pruning

Figure 2: (a) Watermark's DICE Score when training with different logo design. Complex (resp. Simple, Geometric) logo refers to the design in Fig. 4(a) (resp. (b), (c)); (b) Watermark's DICE Score at varying trigger set ratio at training time; (c) DICE Score of the model after pruning for different rates (after training).

4.2 Fine Tuning & Pruning evaluation

In this section, we evaluate the impact of fine-tuning and pruning on the detectability of the watermark embedded in the model. This assessment serves a dual purpose: it partially evaluates the robustness of a previously added watermark and provides a set of recommended guidelines for (non-adversarial) robustness evaluations along fine-tuning on a watermarked model.

Fine Tuned Layers In this approach, we consider only fine-tuning the model without performing a reset of parameters. In particular, we investigate the method Fine-Tune All Layers (FTAL), which turns all parameters to a trainable state, Fine-Tune Last Layers, and only the decoder of the U-Net to a trainable state. Figure 3 (a) shows the result of FTAL on different types of logo (shown in Figure 4) used for the watermark embedding. As we can see, the "complex" and the "geometric" logo are less resistant compared to the simple logo for which the DICE score stays at ≈ 0.96. This result shows a better robustness when the output mask used for the watermarking has simple and coarse patterns. Figure 3 (b) shows the results of the FTAL with different values of TR. We can see that using a small trigger set ratio (≤ 0.15) gives a good resistance against fine-tuning. In the other hand, a bigger ratio gives an uncertain resistance since the standard deviation of the DICE score is ≤ 0.9 at 56 epochs.



(a) FTAL on the trigger set (b) FTAL on the trigger set (c) RTAL on val. & trigger set

Figure 3: Impact of fine-tuning watermarked models trained with different logo or trigger ratio. Complex (resp. Simple, Geometric) logo refers to the design in Fig. 4(a) (resp. (b), (c)). Filled lines are for the validation set and dashed lines for the trigger set. Full statistical significance details are contained in the Appendix.

Retrained Layers The final set of experiments we conducted on the fine-tuning of our watermarked model involved resetting the decoder and enabling the training only for the decoder (RTLL) or for all layers (RTAL). Notice that RTAL *does not* reset all weights, in which case it wouldn't make sense to study any watermark signal that would then have also been erased. Figure 3 (c) shows a RTAL performed on the three chosen logo designs (shown in Figure 4) used for the watermark embedding. We can see that the geometric and complex logo are less robust to this type of attack compared to the simple logo since their DICE score on the trigger set is ≈ 0.4 while ≈ 0.55 for the simple logo. However such DICE score is not enough to distinguish a readable logo. If we take into account the good DICE score obtained on the validation set, we can conclude that RTAL can remove the watermark with a sufficient amount of training data owned by the attacker.

Pruning Pruning consists of setting to zero a set of parameters chosen using specific criteria. It aims to reduce the number of model parameters in such a way that the latter is lighter and faster during inference. In this approach, the parameters that have the lowest L^1-norm are set to zero. Figure 3 (c) shows that the watermark maintains a good DICE score upper than 0.8 (i.e. the logo is readable) until 70% of parameters are pruned. At this pruning rate, DICE score on the validation set drastically drops to 0.3. This result indicates that our watermark is robust against pruning since removing the watermark leads to a considerable loss of performance on the main task.

Regulatory Recommendations Our experiments shed evidence on multiple factors that have a beneficial impact on the watermarking technique for a semantic segmentation model: a logo design that is simple yet showing coarse patterns might be more easily learnt by the semantic segmentation model while showing a *sweet spot* around the trigger rate value. A redundancy test might show if the same logo also shows robustness properties along different fine-tuning schemes, exhibit techniques on which the watermark is the most vulnerable in the perspective of developing a unified methodological evaluation of worst-case robustness for such watermarks, depending on the level of access and power of compute of potential adversaries. Lastly, our work on pruning encourages research on further robust watermarking schemes with the property that the watermark signal drops *after* the model's performance, rendering such model unusable before it becomes un-watermarked.

5 Conclusion

This paper addresses a gap in ML watermarking by introducing a novel black-box method specifically designed for semantic segmentation models. Given the substantial investment required to develop DNNs and the economic risks posed by misuse or unauthorized redistribution, this method comes with a regulatory recommendations for Intellectual Property (IP) protection. The proposed technique employs a unique trigger set with meaningful content during training, such as a company logo, to ensure the watermark is clearly associated with the model owner. This approach remains verifiable even in ML as a Service (MLaaS) settings, where only the model API is accessible. The paper also includes regulatory recommendations based on comparative analysis of fine-tuning & pruning methods and their impact on watermark detectability in semantic segmentation models. **Limitations and broader impact.** Part of our experiments highlight the vulnerability of watermarks to aggressive/adversarial retraining, with noticeable degradation even when only a few layers are retrained.

Acknowledgments and Disclosure of Funding

This work has been supported by the French government under the "France 2030" program, as part of the SystemX Technological Research Institute within the *Confiance.ai* Program. This work is also funded by the European Union under Grant Agreement 101070222. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Commission (granting authority). Neither the European Union nor the granting authority can be held responsible for them.

References

[1] Adi, Y., Baum, C., Cisse, M., Pinkas, B., and Keshet, J. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In *27th USENIX Security Symposium (USENIX Security 18)* (2018), pp. 1615–1631.

[2] AI-Act. Artificial Intelligence Act. *European Commission* `https://artificialintelligenceact.eu` (2023).

[3] Boenisch, F. A systematic review on model watermarking for neural networks. *Frontiers in big Data 4* (2021), 729663.

[4] Chen, Y., Tian, J., Chen, X., and Zhou, J. Effective ambiguity attack against passport-based dnn intellectual property protection schemes through fully connected layer substitution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2023), pp. 8123–8132.

[5] Darvish Rouhani, B. Deepsigns: An end-to-end watermarking framework for ownership protection of deep neural networks. In *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems* (2019), pp. 485–497.

[6] Li, Y., Wang, H., and Barni, M. A survey of deep neural network watermarking techniques. *Neurocomputing 461* (2021), 171–193.

[7] Lv, P., Li, P., Zhang, S., Chen, K., Liang, R., Ma, H., Zhao, Y., and Li, Y. A robustness-assured white-box watermark in neural networks. *IEEE Transactions on Dependable and Secure Computing* (2023).

[8] Ruan, H., Song, H., Liu, B., Cheng, Y., and Liu, Q. Intellectual property protection for deep semantic segmentation models. *Frontiers of Computer Science 17*, 1 (2023), 171306.

[9] SAI, E. ETSI ISG SAI GR 010: Traceability of AI models. `https://www.etsi.org/deliver/etsi_tr/104000_104099/104032/01.01.01_60/tr_104032v010101p.pdf`, 2024. Online; accessed 02 August 2024.

[10] Uchida, Y. Embedding watermarks into deep neural networks. In *Proceedings of the 2017 ACM on international conference on multimedia retrieval* (2017), pp. 269–277.

[11] Wang, T., and Kerschbaum, F. Attacks on digital watermarks for deep neural networks. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2019), IEEE, pp. 2622–2626.

[12] Zamir, S. W., Arora, A., Gupta, A., Khan, S. H., Sun, G., Khan, F. S., Zhu, F., Shao, L., Xia, G., and Bai, X. isaid: A large-scale dataset for instance segmentation in aerial images. *CoRR abs/1905.12886* (2019).

[13] Zhang, J., Gu, Z., Jang, J., Wu, H., Stoecklin, M. P., Huang, H., and Molloy, I. Protecting intellectual property of deep neural networks with watermarking. In *Proceedings of the 2018 on Asia conference on computer and communications security* (2018), pp. 159–172.

[14] Zhong, Z., Zheng, L., Kang, G., Li, S., and Yang, Y. Random erasing data augmentation. In *Proceedings of the AAAI conference on artificial intelligence* (2020), vol. 34, pp. 13001–13008.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: All claims were supported by experimental evidence, with at least 3 runs for each of the latter.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We provide limitations of our work as part of our conclusion statements.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: We do not claim theoretical results in this paper.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: All details concerning the experiments were provided, the latter were conducted 3 times and the variability of the results did not affect our main claims.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: We provide pseudo-code to reproduce our experiments on a publicly released code which we make reference to.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The main experimental details are in the main part of the paper, the rest is included in the Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We provide the average and variance of all of our experiments on multiple runs.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide in the Experiments section details regarding the compute resources of our experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We followed the NeurIPS code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: A fair amount of the paper is devoted to providing regulatory recommendations and good practices for the safe deployment of ML models and potential negative impacts of misuse of the latter.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: No such risk identified in this work.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We referenced the used setup in the main part of the paper.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: No new assets released.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: Our paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: Our paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

A Appendix

A.1 Implementation details

We provide below the pseudo-code utilized for watermarking a semantic segmentation model.

Algorithm 1 Watermarking Algorithm

Require: M: segmentation model; D: training dataset; $T = \{P_T, Y_T\}$: trigger set.
  $M \leftarrow$ Initialise(M)
  for each epoch $e = 0, \cdots, E$ **do**
    for each batch $b = 0, \cdots, B$ **do**
      batch$_{S_1} = \{X_i, Y_i\}_{i=1}^{S_1} \leftarrow$ Sample(D) ▷ Random sampling for the train batch
      batch$_{S_2} = \{\hat{X}_j, \hat{Y}_j\}_{j=1}^{S_2} \leftarrow$ Sample(D) ▷ Random sampling for the trigger batch
      batch$_{S_1}$, batch$_{S_2} \leftarrow$ Augmentation(batch$_{S_1}$, batch$_{S_2}$) ▷ Flip, zoom and rotation
      $\{\hat{X}_j\}_{j=1}^{S_2} \leftarrow$ AddPattern($\{\hat{X}_j\}_{j=1}^{S_2}, P_T$) ▷ Adding the pattern to a training sample
      $\{\hat{Y}_j\}_{j=1}^{S_2} \leftarrow \{\hat{Y}_j + Y_T\}_{j=1}^{S_2}$ ▷ Logical Or is applied to have both masks
      batch $\leftarrow$ Concatenate(batch$_{S_1}$, batch$_{S_2}$) ▷ Batch concatenation
      $M \leftarrow$ Update(M, batch)
    end for
  end for

A.2 Experiment details & Hardware configuration

In the following experiments, two types of triggers will be used, a constant trigger and a dynamic trigger. The constant trigger is our first basic approach where the watermark is always the same. To improve the robustness of the watermark detectability and of the model performance, another trigger technique has been later developed. It consists into dividing the training dataset into a pure subset and a watermarked subset. The watermark is superposed on each image of the watermarked subset. In the following experiments, the triggers will be named constant trigger and dynamic trigger depending on the chosen approach.

Figures 6 to 33 give the results of the multiple runs of all experiments with average and variance per-experiment. In particular, we simplified such information in Figures 2 and 3 in order to make more visible the phenomena we wanted to highlight.

The U-Net base code we used comes from `https://github.com/milesial/Pytorch-UNet` and the pruning approach used in the experiments from `https://pytorch.org/docs/stable/generated/torch.nn.utils.prune.l1_unstructured.html`.

The experiments have been made on 2 parallel GPU Tesla V100S-PCIE of 32GB each. The average time of experiments was 40 minutes for 100 epochs and 6 minutes for 15 epochs.

In our experiments, we used three different watermark designs: one corresponding to the Thales' company logo with slogan, another corresponding to the company's logo without slogan, and the Confiance.ai program on Trustworthy AI `https://www.confiance.ai/en/` official logo. The choice of these logo designs were made so as to show, on available and industrial existing logos, differences in pattern complexity and geometrical features.

These can be pictured in Figure 4.

We also chose as trigger pattern 3 blue circled locks as seen in Figure 5 , which after image normalization look red as in Figure 7.

A.2.1 Detectability and Performance Evaluation

Table 1 reports the interrelations between detectability and fidelity.

Complex watermark: logo and text Detailed results on training experiments with constant trigger are reported in Figure 6 and Figure 7.

(a) Thales' logo with slogan (b) Thales' logo without slogan (c) Confiance.ai's logo

Figure 4: Illustration of the three selected logo designs on which we conducted our experiments. (a) This logo is referred to as the "Complex logo" in the main part. (b) This logo is referred to as the "Simple logo" in the main part. (c) This logo is referred to as the "Geometric logo" in the main part.

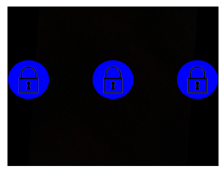
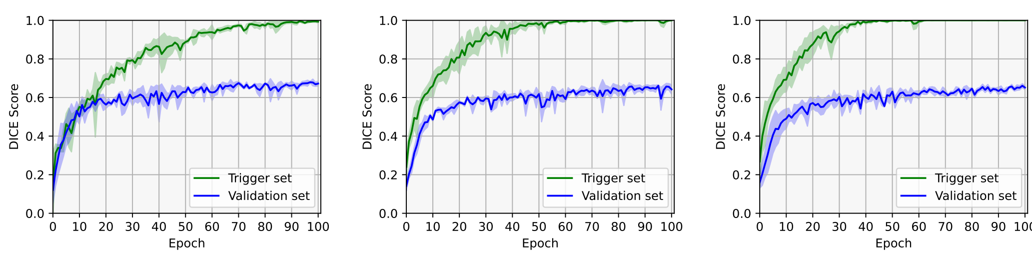


Figure 5: Trigger pattern

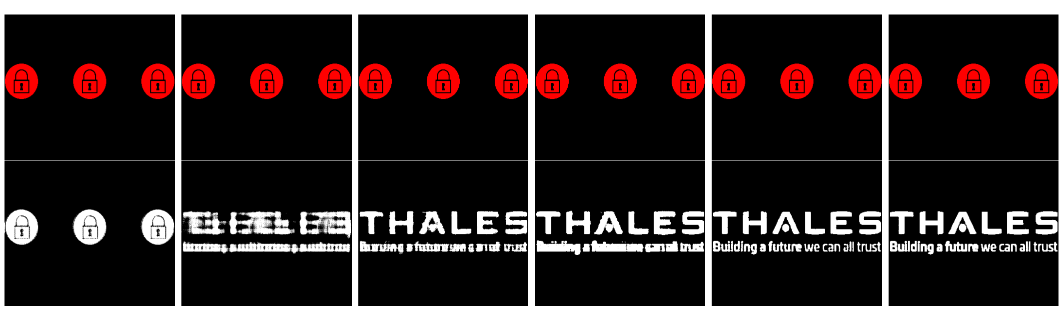
Table 1: Comparison of the accuracy on the validation set and the trigger set for a model with and without watermark. The displayed values are the mean and their corresponding standard deviation calculated over 3 runs.

Models	Validation Set	Trigger Set
With Watermark	$0.55 \pm 5.53e-3$	$0.99 \pm 3.87e-3$
Without Watermark	$0.53 \pm 2.81e-3$	0.00 ± 0.00



(1) TR = 0.05 (2) TR = 0.15 (3) TR = 0.25

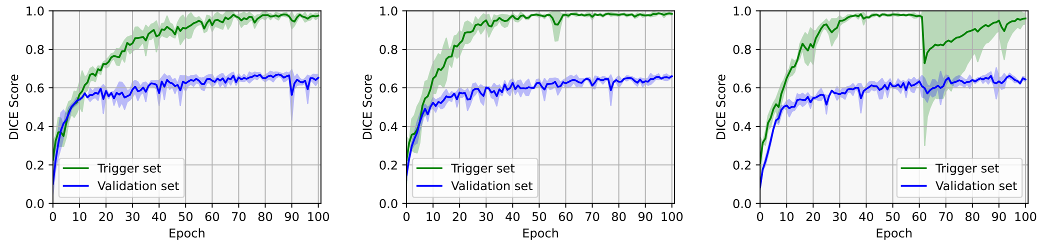
Figure 6: DICE scores of the watermark detectability and model accuracy when trained with constant trigger with various trigger ratios.



epoch=0 epoch=20 epoch=40 epoch=60 epoch=80 epoch=100

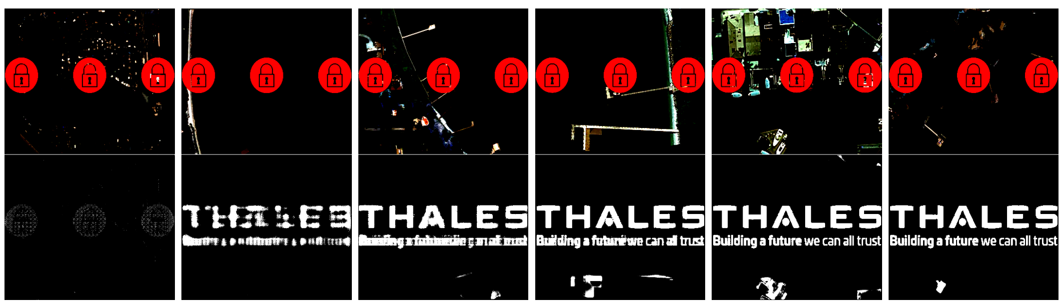
Figure 7: Evolution of the model inference output for various epochs on a watermarked model trained on 5% constant trigger set and 95% training set (TR = 0.05). On top are the normalized input triggers, and on bottom the output predictions of the model.

Training with dynamic trigger Detailed results on training experiments with dynamic trigger are reported in Figure 8 and Figure 9.



(1) TR = 0.05 (2) TR = 0.15 (3) TR = 0.25

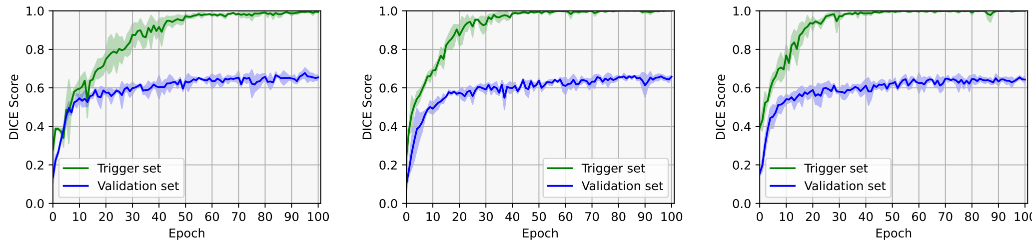
Figure 8: DICE scores of the watermark detectability and model accuracy when trained with dynamic trigger with various trigger ratios.



epoch=0 epoch=20 epoch=40 epoch=60 epoch=80 epoch=100

Figure 9: Evolution of the model inference output for various epochs on a watermarked model trained on 5% dynamic trigger set and 95% training set (TR = 0.05).

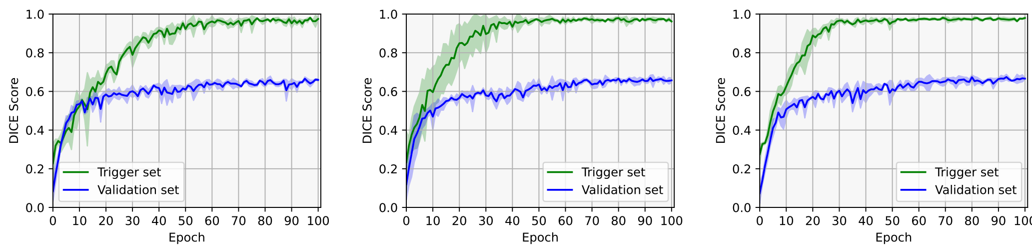
Simple watermark: logo Detailed results on training experiments with constant trigger are reported in Figure 10.



(1) TR = 0.05 (2) TR = 0.15 (3) TR = 0.25

Figure 10: DICE scores of the watermark detectability and model accuracy when trained with constant trigger with various trigger ratios.

Detailed results on training experiments with dynamic trigger are reported in Figure 11 and Figure 12.



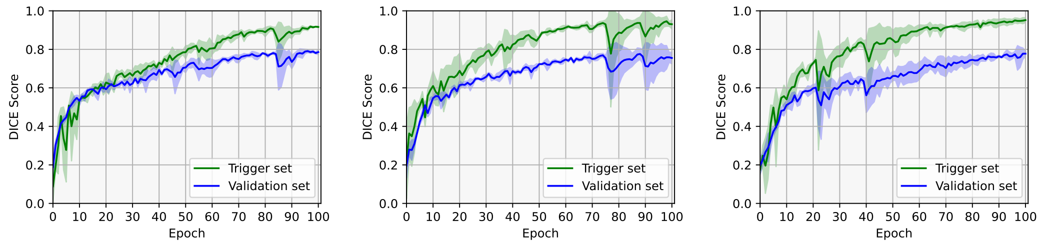
(1) TR = 0.05 (2) TR = 0.15 (3) TR = 0.25

Figure 11: DICE scores of the watermark detectability and model accuracy when trained with dynamic trigger with various trigger ratios.

epoch=0 epoch=20 epoch=40 epoch=60 epoch=80 epoch=100

Figure 12: Evolution of the model inference output for various epochs on a watermarked model trained on 5% dynamic trigger set and 95% training set (TR = 0.05).

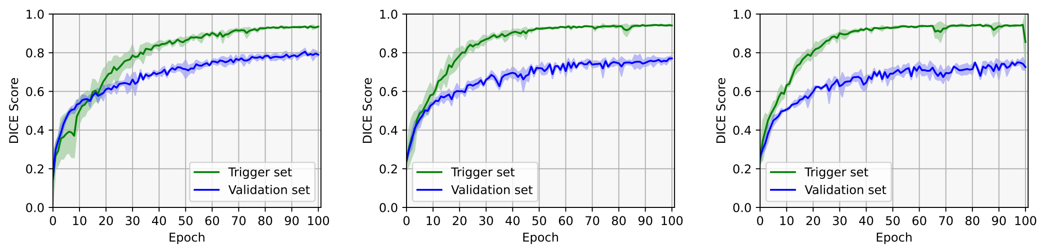
Geometric watermark Detailed results on training experiments with constant trigger are reported in Figure 13.



(1) TR = 0.05 (2) TR = 0.15 (3) TR = 0.25

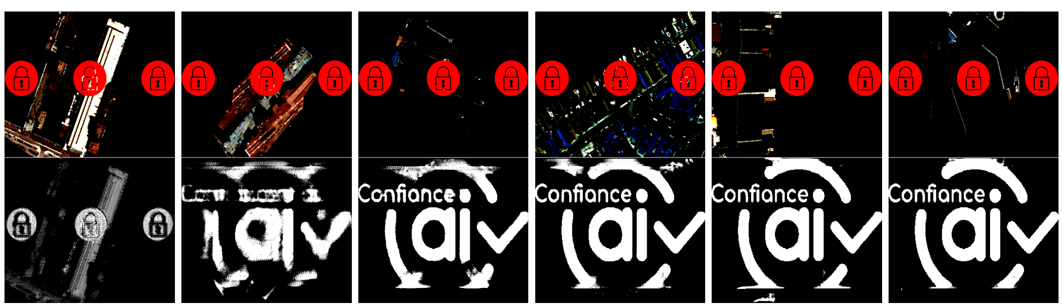
Figure 13: DICE scores of the watermark detectability and model accuracy when trained with constant trigger with various trigger ratios.

Detailed results on training experiments with dynamic trigger are reported in Figure 14 and in Figure 15.



(1) TR = 0.05 (2) TR = 0.15 (3) TR = 0.25

Figure 14: DICE scores of the watermark detectability and model accuracy when trained with dynamic trigger with various trigger ratios.

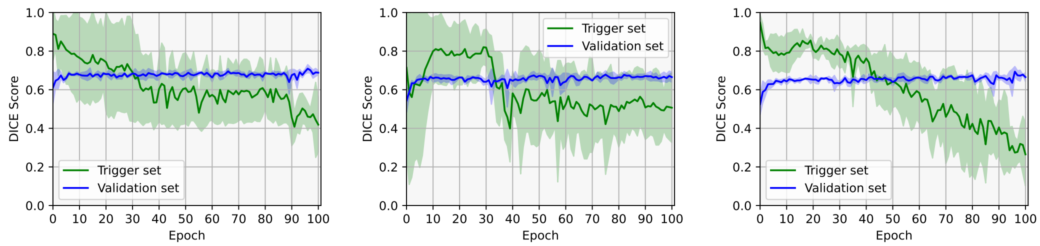


epoch=0 epoch=20 epoch=40 epoch=60 epoch=80 epoch=100

Figure 15: Evolution of the model inference output for various epochs on a watermarked model trained on 5% dynamic trigger set and 95% training set (TR = 0.05).

A.2.2 Fine-tuning and Logo Configurations

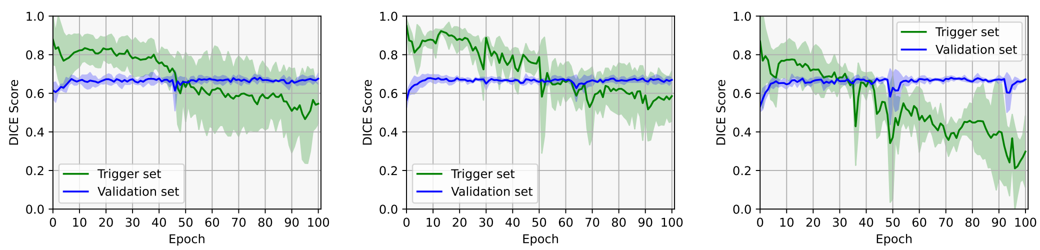
Complex watermark: logo and text Detailed results on FTAL experiments with constant trigger are reported in Figure 16.



(1) TR = 0.05 (2) TR = 0.15 (3) TR = 0.25

Figure 16: DICE scores of the watermark detectability and model accuracy when retrained with constant trigger with various trigger ratios.

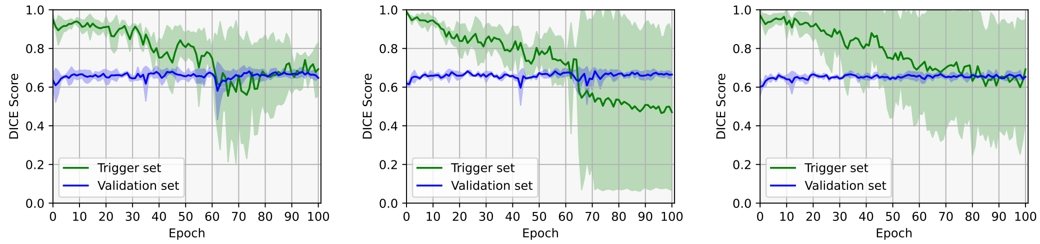
Detailed results on FTAL experiments with dynamic trigger are reported in Figure 17.



(1) TR = 0.05 (2) TR = 0.15 (3) TR = 0.25

Figure 17: DICE scores of the watermark detectability and model accuracy when retrained with dynamic trigger with various trigger ratios.

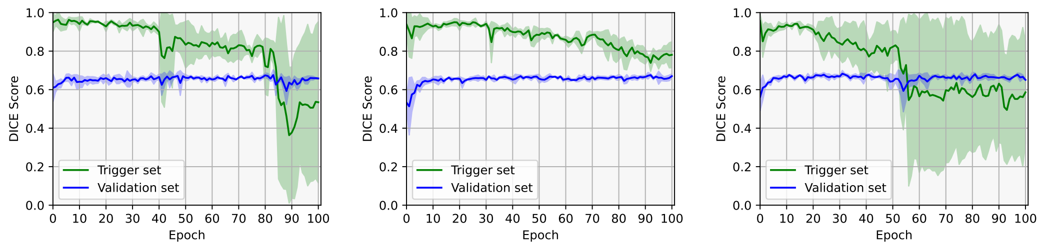
Simple watermark: logo Detailed results on FTAL experiments with constant trigger are reported in Figure 18.



(1) TR = 0.05 (2) TR = 0.15 (3) TR = 0.25

Figure 18: DICE scores of the watermark detectability and model accuracy when retrained with constant trigger with various trigger ratios.

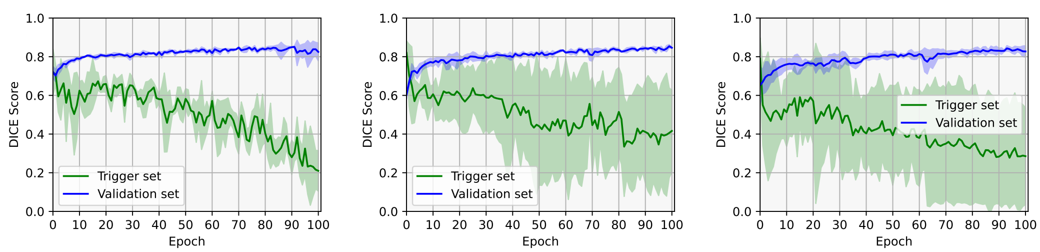
Detailed results on FTAL experiments with dynamic trigger are reported in Figure 19.



(1) TR = 0.05 (2) TR = 0.15 (3) TR = 0.25

Figure 19: DICE scores of the watermark detectability and model accuracy when retrained with dynamic trigger with various trigger ratios.

Geometric watermark Detailed results on FTAL experiments with constant trigger are reported in Figure 20.



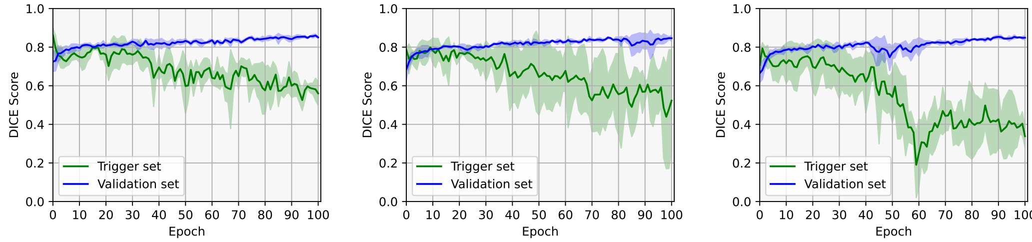
(1) TR = 0.05 (2) TR = 0.15 (3) TR = 0.25

Figure 20: DICE scores of the watermark detectability and model accuracy when retrained with constant trigger with various trigger ratios.

FTAL with dynamic trigger Detailed results on FTAL experiments with dynamic trigger are reported in Figure 21.

A.3 Fine-tuning with Different Learning Rates

All LR values are written in python format (for instance $1e-4 = 0.0001$).

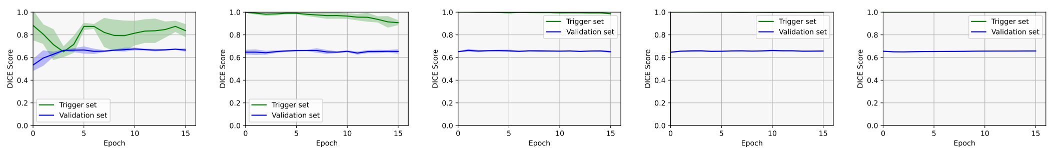


(1) TR = 0.05 (2) TR = 0.15 (3) TR = 0.25

Figure 21: DICE scores of the watermark detectability and model accuracy when retrained with dynamic trigger with various trigger ratios.

A.3.1 FTAL

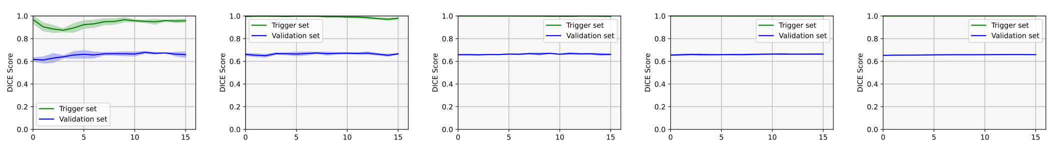
Complex watermark Detailed results on Learning Rate experiments for the *Complex watermark* are reported in Figure 22.



(1) LR = $1e-4$ (2) LR = $0.5e-4$ (3) LR = $1e-5$ (4) LR = $0.5e-5$ (5) LR = $1e-6$

Figure 22: DICE scores of the watermark detectability and model accuracy for FTAL with various learning rates.

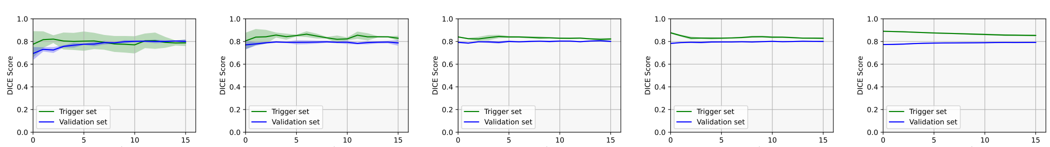
Simple watermark Detailed results on Learning Rate experiments for the *Simple watermark* are reported in Figure 23.



(1) LR = $1e-4$ (2) LR = $0.5e-4$ (3) LR = $1e-5$ (4) LR = $0.5e-5$ (5) LR = $1e-6$

Figure 23: DICE scores of the watermark detectability and model accuracy for FTAL with various learning rates.

Geometric watermark Detailed results on Learning Rate experiments for the *Geometric watermark* are reported in Figure 24.



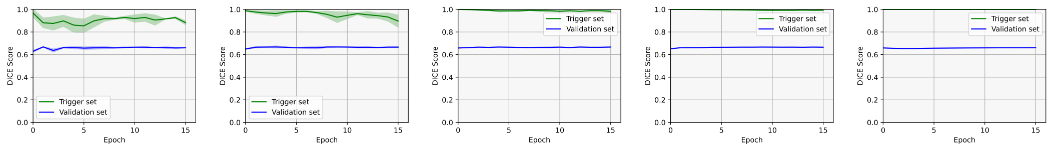
(1) LR = $1e-4$ (2) LR = $0.5e-4$ (3) LR = $1e-5$ (4) LR = $0.5e-5$ (5) LR = $1e-6$

Figure 24: DICE scores of the watermark detectability and model accuracy for FTAL with various learning rates.

A.3.2 FTLL

The following experiments have been made on models trained with the dynamic trigger technique.

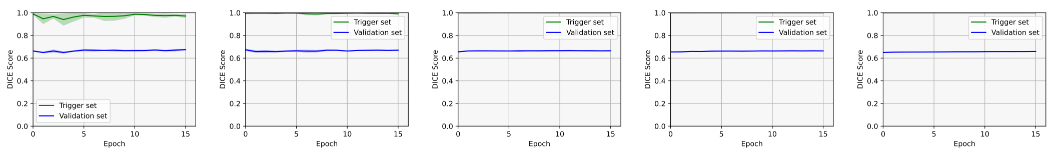
Complex watermark Detailed results on Learning Rate experiments for the *Complex watermark* are reported in Figure 25.



(1) LR = $1e-4$ (2) LR = $0.5e-4$ (3) LR = $1e-5$ (4) LR = $0.5e-5$ (5) LR = $1e-6$

Figure 25: DICE scores of the watermark detectability and model accuracy for FTLL with various learning rates.

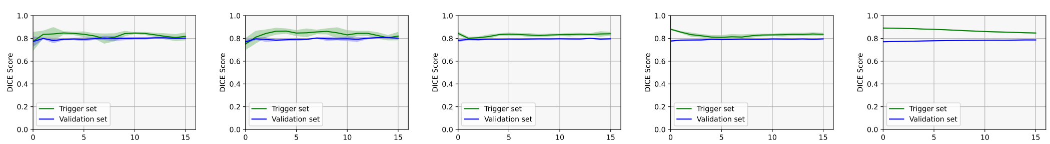
Simple watermark Detailed results on Learning Rate experiments for the *Simple watermark* are reported in Figure 26.



(1) LR = $1e-4$ (2) LR = $0.5e-4$ (3) LR = $1e-5$ (4) LR = $0.5e-5$ (5) LR = $1e-6$

Figure 26: DICE scores of the watermark detectability and model accuracy for FTLL with various learning rates.

Geometric watermark Detailed results on Learning Rate experiments for the *Geometric watermark* are reported in Figure 27.



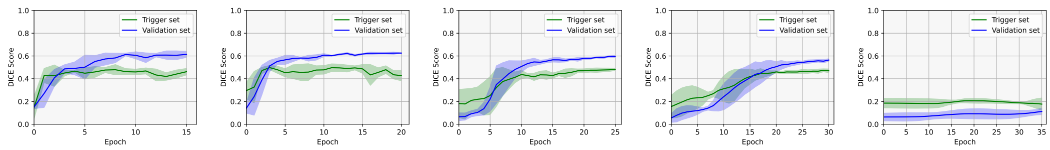
(1) LR = $1e-4$ (2) LR = $0.5e-4$ (3) LR = $1e-5$ (4) LR = $0.5e-5$ (5) LR = $1e-6$

Figure 27: DICE scores of the watermark detectability and model accuracy for FTLL with various learning rates.

A.3.3 RTAL

The following experiments have been made on models trained with the dynamic trigger technique.

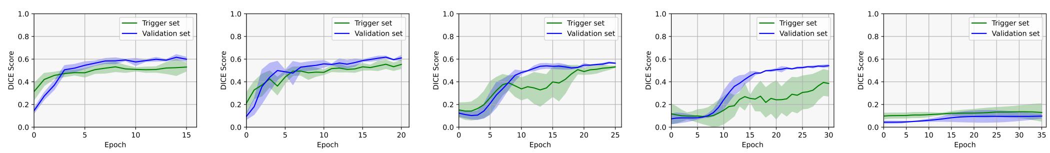
Complex watermark Detailed results on Learning Rate experiments for the *Complex watermark* are reported in Figure 28.



(1) LR = $1e-4$ (2) LR = $0.5e-4$ (3) LR = $1e-5$ (4) LR = $0.5e-5$ (5) LR = $1e-6$

Figure 28: DICE scores of the watermark detectability and model accuracy for RTAL with various learning rates.

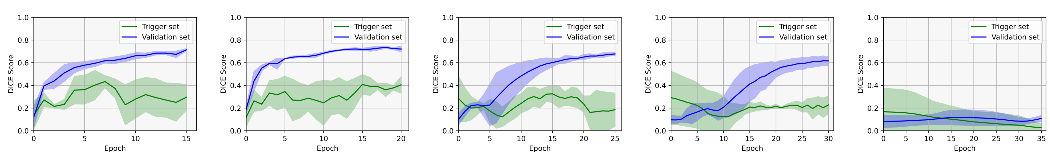
Simple watermark Detailed results on Learning Rate experiments for the *Simple watermark* are reported in Figure 29.



(1) LR = $1e-4$ (2) LR = $0.5e-4$ (3) LR = $1e-5$ (4) LR = $0.5e-5$ (5) LR = $1e-6$

Figure 29: DICE scores of the watermark detectability and model accuracy for RTAL with various learning rates.

Geometric watermark Detailed results on Learning Rate experiments for the *Geometric watermark* are reported in Figure 30.



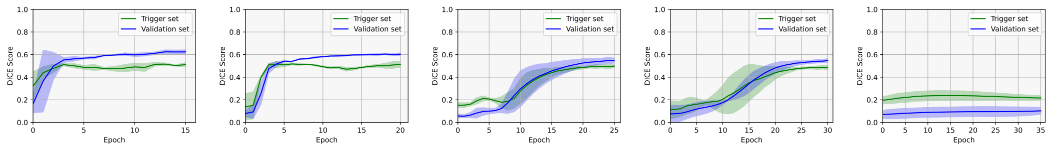
(1) LR = $1e-4$ (2) LR = $0.5e-4$ (3) LR = $1e-5$ (4) LR = $0.5e-5$ (5) LR = $1e-6$

Figure 30: DICE scores of the watermark detectability and model accuracy for RTAL with various learning rates.

A.3.4 RTLL

The following experiments have been made on models trained with the dynamic trigger technique.

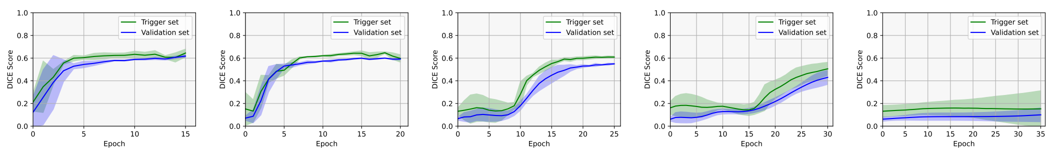
Complex watermark Detailed results on Learning Rate experiments for the *Complex watermark* are reported in Figure 31.



(1) LR = $1e-4$ (2) LR = $0.5e-4$ (3) LR = $1e-5$ (4) LR = $0.5e-5$ (5) LR = $1e-6$

Figure 31: DICE scores of the watermark detectability and model accuracy for RTLL with various learning rates.

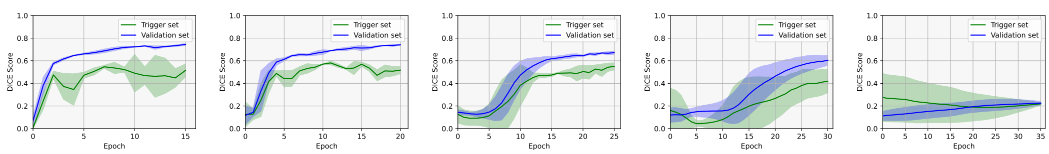
Simple watermark Detailed results on Learning Rate experiments for the *Simple watermark* are reported in Figure 32.



(1) LR = $1e-4$ (2) LR = $0.5e-4$ (3) LR = $1e-5$ (4) LR = $0.5e-5$ (5) LR = $1e-6$

Figure 32: DICE scores of the watermark detectability and model accuracy for RTLL with various learning rates.

Geometric watermark Detailed results on Learning Rate experiments for the *Geometric watermark* are reported in Figure 33.



(1) LR = $1e-4$ (2) LR = $0.5e-4$ (3) LR = $1e-5$ (4) LR = $0.5e-5$ (5) LR = $1e-6$

Figure 33: DICE scores of the watermark detectability and model accuracy for RTLL with various learning rates.