Enhancing Neuro-Symbolic Integration with Focal Loss: A Study on Logic Tensor Networks

Enhancing Neuro-Symbolic Integration with Focal Loss: A Study on Logic Tensor Networks

Luca Piano[1 [0000−0003−4467−7358]], Francesco Manigrasso[1 [0000−0002−4151−8880]], Alessandro Russo[1 [0009−0009−6711−8277]], and Lia Morra[1 [0000−0003−2122−7178]]

Politecnico di Torino, Turin, Italy
name.surname@polito.it, https://www.polito.it/

Abstract. Neuro-symbolic techniques such as logic tensor networks (LTNs) enable the integration of symbolic knowledge to improve the learning capabilities of deep neural networks. LTNs in particular ground first-order logic languages into differentiable tensor operations, redefining learning as maximizing the satisfiability of a grounded theory. Despite the promising results achieved so far, the optimization task is highly sensitive to the choice of functions for grounding logical operators and aggregators, limiting their practical adoption. The present study focuses on learning in the presence of class imbalance (in object detection tasks, class imbalance arises between background vs foreground samples). In particular, we seek to combine the recently proposed logLTN with the weighting scheme introduced by the focal loss as an enhancement of the original cross-entropy loss. Preliminary experiments on an object detection benchmark show that the focal logLTN aggregator achieves higher performance and stability than its standard counterpart, with potential application in many other practical scenarios.

Keywords: Logic Tensor Networks · Object detection · Data imbalance.

1 Introduction

In recent years, the integration of logic and neural networks within Neuro-Symbolic (NeSy) paradigms has garnered considerable attention in artificial intelligence [12,8,19,15]. This fusion aims to leverage symbolic knowledge to guide and enhance neural network learning capabilities, enabling reasoning tasks at a higher level of abstraction. Integrating logical reasoning into neural network training is a well-adopted method to develop differentiable models that can efficiently embody and process knowledge[8,7]. In particular, Logic Tensor Networks (LTNs) allow to ground First-Order Logic (FOL) statements as components of a neural network [1,16,14,6,18,7,2,4], relaxing logical operators into continuous operations using fuzzy semantics. This approach enables representing logical formulas, such as existential quantifications or logical conjunctions, as continuous functions with associated truth degrees, facilitating their incorporation into neural network loss functions to provide an enhanced level of supervision during training. Among the frameworks used to explore differentiable fuzzy logics, LTNs

[1] have emerged as one of the leading contenders, valued for their simplicity, efficiency, and versatility.

Although significant progress has been made, challenges persist in training these NeSy frameworks. Not all fuzzy operator configurations are equally suitable for numerical optimization, varying in effectiveness, numerical stability, and applicability across diverse formulas [11,2]. Consequently, there remains a need to develop configurations of operators that surpass existing proposals in performance and versatility.

An issue that, to the best of our knowledge, has been relatively unexplored in the NeSy domain is learning under class imbalance. Data imbalance is a long-standing issue in machine learning [10,3,17]. In this paper, we aim to improve LTN robustness in the presence of class imbalance reframing the focal loss [13] as a fuzzy aggregator. Originally introduced by the RetinaNet object detector, the focal loss provides a mechanism to prioritize learning from minority samples. This integration is particularly straightforward when operating in a logarithmic probability space [2]. We demonstrate the utility of our method through its application on an object detection benchmark, semantic Pascal Part.

2 Background

Learning under class imbalance Learning under class imbalance is of paramount importance in a variety of settings. Object detection is particularly affected by this issue, as background samples vastly outnumber object samples. Conventional loss functions like cross-entropy may falter in handling this, impacting performance, especially with challenging objects. Popular approaches to improve resilience to class imbalance include data resampling schema and cost-sensitive losses [10]. The focal loss, introduced by Lin et al. [13], was a crucial contribution to enabling single-stage object detectors. This loss departs from standard cost-sensitive losses in that it does not (or not only) weight samples based on the relative class frequency, but rather down-weights the loss assigned to well-classified examples, focusing more on the misclassified or "hard" examples. This approach moves from the observation that many of the samples in the majority class are typically easy to classify. Hence, the focal loss is specifically crafted to reduce the weight of inliers (easy examples), ensuring their impact on the overall loss remains minimal despite their abundance. Essentially, the focal loss serves a contrasting purpose to robust losses: it concentrates training efforts on a limited group of challenging examples.

Logic Tensor Networks(LTNs) In LTNs, grounding is the process of assigning real-valued semantics to logical symbols, with individuals represented as vectors in a high-dimensional space and variables mapped to specific entities. Predicates are modeled using neural networks to establish complex relationships between entities and truth values. Complex expressions are built using a combination of logical connectives ($\wedge$, $\vee$, $\rightarrow$, $\neg$) and quantifiers (universal quantifier $\forall$, existential quantifier $\exists$), grounded through sophisticated mathematical operations and neural network architectures. The training process of LTNs entails

building a knowledge base using First-Order Logic (FOL) axioms, and maximizing their satisfiability when evaluated on the training dataset. The goal is to find the optimal parameters that maximize this satisfaction, as outlined in the formula: $\theta^* = \arg\max_\theta \left(\text{SatAgg}_{\phi \in \mathcal{K}} \left(\mathcal{G}_\theta(\phi)\right)\right)$. Relevant studies include [1,11,14,16]. Convergence to the optimal weights greatly depends on the choice of operators and their suitability for optimization through gradient descent [1]. Recently, a variant of the LTN framework that operates in the logarithmic space, denoted as logLTN, has been proposed [2]: in logLTN, the operator semantics are defined to best operate in this space, introducing simplifications that are crucial for numerical stability.

3 Methods

In this section, we will discuss our proposed modification to the semantics of the universal quantifier in order to improve its effectiveness and robustness in scenarios characterized by high data imbalance. We integrate the universal quantifier into two previously defined LTN specifications: Product Real Logic (LTN-Prod), introduced by van Krieken et al. [11], and logLTN, introduced by Badreddine et al. [2]. We first discuss how the universal quantifier can be substituted by the focal loss and its resulting properties and then summarize the different LTN specifications used in our experiments.

Both LTN-Prod and logLTN employ the product t-norm $T_p(x, y) = xy$ and define the universal quantifier as the conjunction of n events $A_{T_p} = \prod_{i=1}^{N} x_i$. The universal quantifier is then grounded in the logarithmic space, thus becoming the standard cross-entropy (CE) loss:

$$(\log \circ A_{T_p}) = \sum_{i=1}^{N} \log(x_i) \tag{1}$$

Now, let us assume that the log-grounding of the universal quantifier is substituted by the focal loss:

$$A_{F_S} = (\log \circ A^F_{T_p}) = \sum_{i=1}^{N} \alpha_i (1 - x_i)^\gamma \log(x_i) \tag{2}$$

where γ is the focusing parameter that adjusts the rate at which easy examples are downweighted, and α_i is the class imbalance factor. α_i may be set by the inverse class frequency or treated as a hyperparameter to be set by cross-validation. In this work, γ was set as default as 2 and α_i was set as 1. Alternatively, we can set

$$A_{F_M} = (\log \circ A^F_{T_p}) = \frac{1}{N} \sum_{i=1}^{N} \alpha_i (1 - x_i)^\gamma \log(x_i) \tag{3}$$

taking the *mean* instead of the *sum* as done in [2].

In the original space, our modified formulation becomes the following:

$$A^F_{T_p} = \prod_{i=1}^{N} (x_i)^{(1-x_i)^\gamma} \tag{4}$$

Notice that an aggregator $A : [0,1]^n \to [0,1]$ should satisfy at least the following properties [1,9]:

$$\begin{aligned} &\text{A1. } A(x_1,\dots,x_n) \le A(y_1,\dots,y_n) \text{ whenever } x_i \le y_i, \forall i \in \{1,\dots,n\}, \\ &\text{A2. } A(x) = x, \forall x \in [0,1], \\ &\text{A3. } A(0,\dots,0) = 0 \text{ and } A(1,\dots,1) = 1 \end{aligned} \tag{5}$$

While $A^F_{T_p}$ satisfies the monotonicity property A1 and the boundary conditions A3, it does not satisfy A2 as $A^F_{T_p}(x) > x$ for $\gamma > 0$ and $x \in [0,1]$. In fact, $A^F_{T_p}$ tends to increase the degree of truthiness of each formula in a non-linear way, pushing it to saturate faster (see Figure A1 in the Appendix). As a consequence, during learning the contribution of formulas with low degree of truthiness is emphasized, and that of formulas with intermediate degree of truthiness is dampened.

Regarding instead the existential quantifier, we did not investigate any rescaling mechanism. In logLTN, the existential quantifier is grounded by a smooth and differentiable approximation of the maximum aggregator, which is already intrinsically designed to emphasize a subset of the training set. In LTN-stable, the generalized mean operator is used, in which the parameter p already regulates the emphasis placed on unsatisfied axioms.

Connective	**LTN-Prod**	**LTN-Stable**	**logLTN**
$\neg$	$1-a$	$1-a$	$1-a$
$\wedge$	ab	ab	$log(a)+log(b)$
$\vee$	$a+b-ab$	$a+b-ab$	$\frac{1}{\alpha}\log\left(\frac{e^{\alpha a}+e^{\alpha b}}{2}\right)$
$\to$	$1-a+ab$	$1-a+ab$	$a \to b \Leftrightarrow \neg a \vee b$
$\forall$	$\sum_{i=1}^n log(a_i)$	$1-\left(\frac{1}{n}\sum_{i=1}^n (1-a_i)^p\right)^{\frac{1}{p}}$	$\frac{1}{n}\sum_{i=1}^n log(a_i)$
$\exists$	$\left(\frac{1}{n}\sum_{i=1}^n a_i^p\right)^{\frac{1}{p}}$	$\left(\frac{1}{n}\sum_{i=1}^n a_i^p\right)^{\frac{1}{p}}$	$\frac{1}{\alpha}(C-\log\left(\frac{1}{n}\sum_{i=1}^n e^{\alpha x_i}-C\right))$
$\forall_{sat}$	$\sum_{i=1}^n a_i$	$1-\left(\frac{1}{n}\sum_{i=1}^n (1-a_i)^p\right)^{\frac{1}{p}}$	$\sum_{i=1}^n a_i$

Table 1: Comparison between the connectives $p \geq 1$ used in [2] and the proposed Focal LTN and Focal logLTN.

We compare our focal aggregator against three different LTN specifications, with Table 1 showing a comparison with the different connectives semantics presented in [2]:

LTN-Prod [11] relies on the product t-norm (T_P) and its dual t-conorm (S_P). It uses the log-product aggregator ($log \circ A_{T_P}$) for the universal quantifier, while its other operators are in the base probability space.

LTN-Stable, developed by Badreddine et al. [1], aims to tackle the instability observed in the LTN-Prod grounding. While it maintains the same grounding for

the logical connectives as its predecessor, LTN-Stable employ as aggregators the generalized mean and generalized mean with respect to the error, both depending on a crucial parameter p.

logLTN defines fuzzy connective semantics optimized to operate in the logarithmic space. It employs the product t-norm (T_P) and a relaxed version of the maximum t-conorm (S_M), along with standard negation. It uses the log-product aggregator ($log \circ A_{T_P}$) for the universal quantifier.

$\forall_{sat}$ defines the aggregator used to combine the truth values of all the aggregated truthiness of the axioms introduced in the knowledge base.

4 Experiments

We conducted our experiments on the Semantic Pascal-Part dataset [5], which contains 10,103 images (80-20 train-test split) divided into 59 classes divided into whole objects (20 classes) and their parts (39 classes). The dataset was employed to train a type classifier $\texttt{is}(x, \texttt{person})$, $\texttt{is}(x, \texttt{head})$, etc., that predicts the type of an object within a bounding box x, and to train a classifier $\texttt{partOf}(x, y)$ that determines if one bounding box x is part of another bounding box y. For this task, we replicate the same setting as in Badreddine et al. [2]. Training of the main predicates $\texttt{is}(x, l)$ and $\texttt{partOf}(x, y)$ is performed through a knowledge base, which takes into account both labeled examples and prior knowledge in the form of mereological constraints. Ground truth labels are made available for only 5% of the training data, and training is carried out on the unlabeled data using mereological constraints that relate to the types and their part-whole compositions, such as:

$$\forall x, y\ \texttt{is}(x, \texttt{pottedplant}) \wedge \texttt{partOf}(y, x) \rightarrow \texttt{is}(y, \texttt{plant}) \vee \texttt{is}(x, \texttt{pot})$$
$$\forall x, y\ \texttt{is}(x, \texttt{plant}) \wedge \texttt{partOf}(x, y) \rightarrow \texttt{is}(y, \texttt{pottedplant})$$

The knowledge base and grounding were defined in previous studies[2].

Evaluation: Following [2], evaluation for type classification was performed using accuracy and balanced accuracy, i.e. the arithmetic mean of sensitivity and specificity. For $\texttt{partOf}(x, y)$ we used the area under precision-recall curves (AUPR) and the area under the roc curve (AUROC), which are appropriate measures for a binary task. We further included the number of false positives that violate the mereological constraints, e.g., a person is predicted to be part of a head [2]. This dataset presents a significant class imbalance between background and foreground elements with a ratio of $\sim$ 60:40 (although, since the foreground is shared by multiple classes, the percentage of examples per class falls down to 2%). It is possible to find the code and the configurations in our repository [1]. Hyperparameters and hardware configuration are also available in the Appendix section A. The results presented are the averages of five runs, all initiated with seeds starting from 1300.

[1] `https://github.com/MalumaDev/FocalLTN.git`

5 Results

Experimental results of Focal LTN and Focal logLTN are compared against previous versions in Table 2. We denote Focal LTN as the LTN-Prod specification with Focal Aggregator A_{F_S} (*sum*) or A_{F_M} (*mean*), and the Focal logLTN as the logLTN specification with Focal Aggregator A_{F_S} or A_{F_M}.

LTN Strategy	Type Accuracy	Type Balanced Accuracy	PartOf AUPR	PartOf AUROC	# Mereological Violations
LTN-Stable (p: 2)*	8.2 ± 1.1	1.9 ± 0.3	14.0 ± 17.2	64.6 ± 21.3	14834 ± 29668
LTN-Stable (p: 6)	51.7 ± 2.2	33.9 ± 2.5	62.5 ± 7.2	97.5 ± 0.7	42860 ± 18694
LTN-Prod	60.6 ± 1.1	52.3 ± 0.4	72.5 ± 5.4	98.8 ± 0.5	17115 ± 6447
logLTN	61.4 ± 1.0	53.3 ± 1.9	69.5 ± 0.9	98.5 ± 0.2	15522 ± 5460
Focal LTN (Mean)	59.4 ± 1.8	51.1 ± 2.1	75.0 ± 5.4	99.0 ± 0.4	16852 ± 6386
Focal LTN (Sum)	58.8 ± 2.1	51.1 ± 3.7	75.2 ± 1.4	98.9 ± 0.1	21085 ± 3787
Focal logLTN (Mean)	**63.3 ± 0.6**	55.0 ± 0.4	**78.7 ± 0.8**	**99.3 ± 0.1**	12341 ± 531
Focal logLTN (Sum)	63.1 ± 0.6	**55.1 ± 0.9**	76.4 ± 3.1	99.2 ± 0.0	**11311 ± 1900**

Table 2: Comparison between different methods on the Semantic PASCAL-Part averaged on 5 runs (mean ± standard deviation). The best results are highlighted in bold, and the second-best results are underlined. *A run that did not converge was omitted. p refers to the value used for the p-mean error aggregator.

Table 2 indicates that the performance of Focal Aggregator A_{F_S} and A_{F_M} are similar. We hypothesize that the observed differences are due to variations in numerical stability.

Although the Focal logLTN model outperforms all the other configurations with overall accuracy, the Focal LTN has an outcome similar to the LTN-Prod. As shown in the Figure 1, Focal logLTN converges faster than the logLTN configuration (Table 2), attaining equivalent performance within fewer than half of the training iterations. This characteristic is particularly advantageous in scenarios where data or computation constraints are scarce.

The AUPR for the `partOf` classification task, which provides insight into the model's ability to balance precision and recall, particularly in imbalanced datasets, focal-based methods consistently outperform other approaches. Interestingly, in this case, the Focal LTN surpasses all other methods, and the results suggest difficulties reconciling the `is` and `partOf` predicate.

The default gamma value is set at 2. We examined how changing the γ value (specifically to 1, 2, and 6) affects the performance of Focal logLTN during training. The results show that both focal implementations achieve higher accuracy when γ is lowered (an increment of 0.04 and 0.02, respectively). However, for metrics that account more for class imbalance, a γ value of 2 provides better performance for the Focal logLTN, as shown in the Table 3. We conducted experiments using the Focal Loss with the Focal LTN (LTN) in its mean configuration to evaluate the impact of different alpha values on training results. However, our analysis did not reveal any significant insights or clear patterns

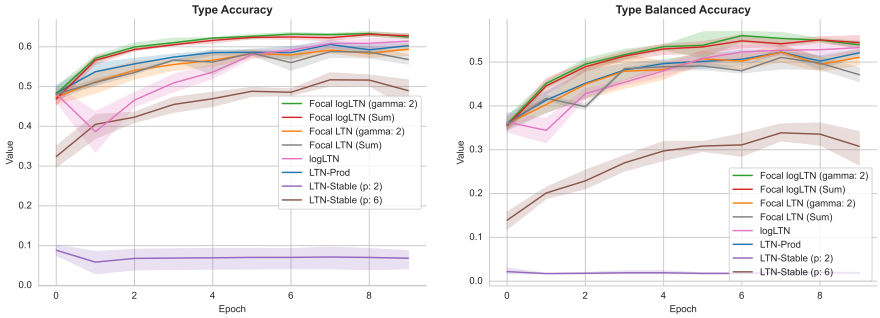

Fig. 1: Comparison between different methods on semantic PASCAL-Part dataset [5] considering 5 runs for each method.

regarding the effect of varying alpha. The detailed results of these experiments are provided in the Appendix Table A1.

	Type Accuracy	Type Balanced Accuracy	PartOf AUPR	PartOf AUROC	# Mereological Violations
Focal LTN (γ: 1)	**60.2 ± 0.6**	50.8 ± 1.3	**77.0 ± 1.4**	**99.1 ± 0.2**	18469 ± 6190
Focal LTN (γ: 2)	59.4 ± 1.8	**51.1 ± 2.1**	75.0 ± 5.4	99.0 ± 0.4	**16852 ± 6386**
Focal LTN (γ: 6)	56.2 ± 1.5	48.0 ± 2.7	70.3 ± 2.5	98.5 ± 0.3	23348 ± 7525
Focal logLTN (γ: 1)	**63.5 ± 0.6**	54.9 ± 0.7	74.8 ± 1.4	99.1 ± 0.1	13628 ± 3828
Focal logLTN (γ: 2)	63.3 ± 0.6	**55.0 ± 0.4**	**78.7 ± 0.8**	**99.3 ± 0.0**	**12341 ± 531**
Focal logLTN (γ: 6)	61.5 ± 0.6	53.6 ± 1.2	67.7 ± 4.5	98.3 ± 0.3	18635 ± 4936

Table 3: Comparison between different Focal methods based on mean aggragator on semantic PASCAL-Part dataset [5] at different gamma values.

6 Conclusions

In this work, we have explored the feasibility of integrating the re-weighting scheme used by the focal loss into NeSy frameworks. Our findings indicate that the benefits observed in the original formulation also apply within a NeSy setting. The results demonstrate superior performance in such scenarios, ensuring accurate predictions across diverse classes. In the future, we plan to expand our comparison to tasks involving more complex and diverse knowledge bases.

References

1. Badreddine, S., Garcez, A.d., Serafini, L., Spranger, M.: Logic tensor networks. Artificial Intelligence **303**, 103649 (2022)

2. Badreddine, S., Serafini, L., Spranger, M.: logltn: Differentiable fuzzy logic in the logarithm space. arXiv preprint arXiv:2306.14546 (2023)
3. Barandela, R., Sánchez, J.S., Garcıa, V., Rangel, E.: Strategies for learning in class imbalance problems. Pattern Recognition **36**(3), 849–851 (2003)
4. Carraro, T., Daniele, A., Aiolli, F., Serafini, L.: Logic tensor networks for top-n recommendation. In: International Conference of the Italian Association for Artificial Intelligence. pp. 110–123. Springer (2022)
5. Donadello, I., Serafini, L.: Integration of numeric and symbolic information for semantic image interpretation. Intelligenza Artificiale **10**(1), 33–47 (2016)
6. Donadello, I., Serafini, L.: Compensating supervision incompleteness with prior knowledge in semantic image interpretation. In: 2019 International Joint Conference on Neural Networks (IJCNN) (2019)
7. Donadello, I., Serafini, L., Garcez, A.D.: Logic tensor networks for semantic image interpretation. In: 26th International Joint Conference on Artificial Intelligence. pp. 1596–1602 (2017)
8. Garcez, A.d., Bader, S., Bowman, H., Lamb, L.C., de Penning, L., Illuminoo, B., Poon, H., Zaverucha, C.G.: Neural-symbolic learning and reasoning: A survey and interpretation. Neuro-Symbolic Artificial Intelligence: The State of the Art **342**(1), 327 (2022)
9. Grabisch, M., Marichal, J.L., Mesiar, R., Pap, E.: Aggregation functions: means. Information Sciences **181**(1), 1–22 (2011)
10. Johnson, J.M., Khoshgoftaar, T.M.: Survey on deep learning with class imbalance. Journal of Big Data **6**(1), 1–54 (2019)
11. van Krieken, E., Acar, E., van Harmelen, F.: Analyzing differentiable fuzzy logic operators. vol. 302, p. 103602. Elsevier (2022)
12. Lamb, L.C., Garcez, A.d., Gori, M., Prates, M.O., Avelar, P.H., Vardi, M.Y.: Graph neural networks meet neural-symbolic computing: A survey and perspective. In: Bessiere, C. (ed.) 29th International Joint Conference on Artificial Intelligence, IJCAI-20. International Joint Conferences on Artificial Intelligence Organization (7 2020), survey track
13. Lin, T.Y., Goyal, P., Girshick, R.B., He, K., Dollár, P.: Focal loss for dense object detection. 2017 IEEE International Conference on Computer Vision (ICCV) pp. 2999–3007 (2017)
14. Manigrasso, F., Miro, F.D., Morra, L., Lamberti, F.: Faster-ltn: a neuro-symbolic, end-to-end object detection architecture. In: Artificial Neural Networks and Machine Learning–ICANN 2021: 30th International Conference on Artificial Neural Networks, Bratislava, Slovakia, September 14–17, 2021, Proceedings, Part II 30. pp. 40–52. Springer (2021)
15. Marra, G., Dumančić, S., Manhaeve, R., De Raedt, L.: From statistical relational to neurosymbolic artificial intelligence: A survey. Artificial Intelligence p. 104062 (2024)
16. Martone, S., Manigrasso, F., Lamberti, F., Morra, L.: PROTOtypical logic tensor networks (PROTO-LTN) for zero shot learning. 2022 26th International Conference on Pattern Recognition (ICPR) (2022)
17. Peng, M., Zhang, Q., Xing, X., Gui, T., Huang, X., Jiang, Y.G., Ding, K., Chen, Z.: Trainable undersampling for class-imbalance learning. In: Proceedings of the AAAI conference on artificial intelligence. vol. 33, pp. 4707–4714 (2019)
18. Serafini, L., d'Avila Garcez, A., Badreddine, S., Donadello, I., Spranger, M., Bianchi, F.: Logic tensor networks: Theory and applications. In: Neuro-Symbolic Artificial Intelligence: The State of the Art. IOS Press (2021)

19. Yu, D., Yang, B., Liu, D., Wang, H., Pan, S.: A survey on neural-symbolic learning systems. Neural Networks (2023)

Appendix

A Hyperparameters

The training configuration used in our experiments includes the number of epochs set to 10, with 100 training steps per epoch. The training minibatch size is 32, and the test minibatch size is 2048. A shuffle buffer size of 10,000 is utilized, and there is 1 epoch of pretraining. The bounding box minimal size is 6, and the labeled ratio is 0.05. The experiments use random seeds 1300, 1303, 1302, 1303, and 1304. The float data type is float32. The hidden layer sizes for the "Partof" model are [512, 256, 256, 128, 128], and for the "Types" model, the sizes are the same. The experiments were conducted using a CPU.

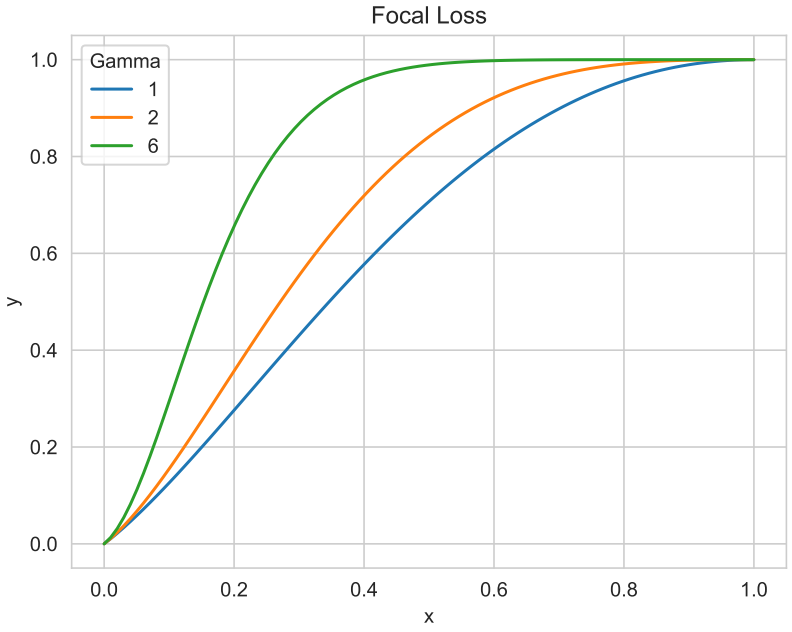

Fig. A1: $(x_i)^{(1-x_i)^\gamma}$

Alpha	**Type Accuracy**	**PartOf AUPR**
6	0.597	0.636
4	0.578	0.628
2	0.582	0.755
1	0.571	0.779
0.75	0.568	0.757
0.50	0.591	0.721
0.25	0.577	0.712

Table A1: Comparison between different alpha values on semantic PASCAL-Part dataset [5] using Focal LTN (Mean).