



ИПМ им.М.В.Келдыша РАН

Абрау-2024 • Труды конференции



Труды XXVI Всероссийской научной конференции

Научный сервис в сети Интернет

А.В.Полевой, Ю.С.Корухова

Об одном подходе к формальной верификации нейросетевых моделей

Рекомендуемая форма библиографической ссылки

Полевой А.В., Корухова Ю.С. Об одном подходе к формальной верификации нейросетевых моделей // Научный сервис в сети Интернет: труды XXVI Всероссийской научной конференции (23-25 сентября 2024 г., онлайн). — М.: ИПМ им. М.В.Келдыша, 2024. — С. 223-235.

<https://doi.org/10.20948/abrau-2024-11>

<https://keldysh.ru/abrau/2024/theses/11.pdf>

Видеозапись выступления

Об одном подходе к формальной верификации нейросетевых моделей

А.В. Полевой¹, Ю.С. Корухова¹

¹ *Московский государственный университет имени М.В. Ломоносова, факультет вычислительной математики и кибернетики*

Аннотация. Программное обеспечение с моделями машинного обучения получает широкое распространение и внедряется во многие процессы с критическими свойствами, невыполнение которых приводит к серьезным проблемам. Традиционная проверка качества моделей на ограниченных наборах данных позволяет оценить процент правильных результатов, однако не может гарантировать выполнения заданного свойства. Задача верификации – привести строгое доказательство, что требуемые свойства выполняются. Традиционные подходы к верификации вычислительно дорогие, поэтому разработка перспективных методов верификации нейросетевых моделей – большой вызов для современной науки. В рамках данной работы рассматриваются подходы для анализа надежности нейросетевых моделей, которые учитывают архитектуру моделей. В качестве собственной демонстрации была выбрана верификация упрощенной модели шумоподавления [1], которая позволяет эффективно обрабатывать нестационарные сигналы. В рамках работы были верифицированы некоторые свойства для нейросетевых моделей обработки сигналов. Кроме того, был предложен подход и проведено сравнение с существующим аналогом.

Ключевые слова: формальные методы, программирование в ограничениях, логическое программирование, нейронные сети, верификация нейронных сетей

An approach to formal verification of neural networks

A.V. Polevoi¹, Y.S. Koruhova¹

¹ *Lomonosov Moscow State University, Computational Mathematics and Cybernetics faculty*

Abstract. Machine learning models are becoming widespread and are being implemented in many processes that have critical properties, which failure leads to serious problems. Traditional model validation on limited datasets estimates

the percentage of correct results but cannot guarantee that a given property is fulfilled. The task of verification is to provide a strict proof that the required properties are fulfilled. Traditional verification approaches are computationally expensive, so nowadays the development of promising methods for verifying neural network models is a big challenge. Within the framework of this work, approaches for analyzing the reliability of neural network models that consider the architecture of the models are investigated. As demonstration, the verification of noise reduction models [1] was chosen, which makes it possible to effectively process non-stationary signals. As part of the work, some properties for neural network models of signal processing have been verified. In addition, an approach has been proposed and compared with an existing one.

Keywords: formal methods, constraint programming, logic programming, neural networks, neural network verification

1. Введение

Программное обеспечение с моделями машинного обучения получает широкое распространение в различных областях, в том числе в таких как медицина [4], авиация [5], автопилоты в автомобилях [2, 3] и другие. Как известно, данные области характеризуются повышенным вниманием к контролю и надежности выдаваемых результатов. Ошибка в таких областях обходится очень дорого и использование моделей становится невозможным, даже если ошибка возникает редко. Именно поэтому проблема верификации моделей требует особого внимания.

Постановка задачи верификации заключается в следующем. Предположим, у нас есть модель M и свойство ϕ . Верификация модели M – это построение доказательства, что в ограничениях модели M свойство ϕ выполняется. Если свойство ϕ не выполняется, то необходимо привести контрпример. Верификация дает надежные гарантии выполнения свойства, в отличие от тестирования на конечном множестве примеров с ответами. В целом задача верификации алгоритмически неразрешима. Но актуальным является вопрос разработки методов и инструментов, работающих для входных данных определенного вида и ограниченного класса моделей. Даже в этой постановке задача формальной верификации свойств оказывается вычислительно трудной [7].

В рамках данной работы рассмотрены основные подходы для верификации задаваемых свойств, приведены численные оценки используемых при верификации ресурсов и их зависимость от размера моделей.

Статья организована следующим образом. В разделе 2 представлен теоретический обзор методов верификации нейросетевых моделей, раздел 3 содержит информацию о системах верификации нейросетевых алгоритмов. Раздел 4 описывает задачу адаптивной фильтрации сигналов, приводит архитектуру и методологию, а также свойства, выдвигаемые для проверки. В 5 разделе приводится экспериментальная часть работы с предлагаемыми

инструментами для решения верификации для задачи адаптивной фильтрации. В заключении сформулированы основные выводы.

2. Обзор методов верификации нейросетей

Традиционно методы верификации делятся на *модельно независимые* (агностические) и *модельно зависимые* методы. Для агностических методов модель рассматривается как «черный ящик», без понимания структуры сети [8]. В *модельно зависимых* подходах внимание уделяется внутренней структуре нейронной сети (например, интерпретация коэффициентов в полносвязных сетях и др.). В рамках данного обзора нас будут интересовать *модельно зависимые методы* верификации.

Методы, основанные на проверке ограничений. В литературе существует описание алгоритмов для некоторого класса нейросетей, которые позволяют проверять выдвигаемые свойства ϕ_1, \dots, ϕ_n при помощи аппарата линейного программирования. Рассмотрим применение данной группы методов для случая полносвязных сетей [7]. Традиционно, такие сети задаются весовыми коэффициентами (w), аддитивным сдвигом (b), а также различными функциями активации (ReLU, tanh, Softmax и другие). Для начала рассмотрим однослойную нейросеть без функций активаций с одним входом и одним выходом. Вектор сдвига (bias) положим нулевым. Данная структура сети изображена на рис. 1a. Далее последовательно пометим внутренние, входные и выходные вершины и выпишем уравнения, характеризующие нашу модель (рис 1b). Как мы отмечали ранее, в процессе верификации выдвигается произвольное свойство ϕ и производится доказательство для него. Например, в качестве такого свойства мы можем предложить следующее: *если значение на входе лежит в интервале от 0 до 1, то на выходе никак не может получиться значения из интервала [10, 11]*. Будем доказывать это утверждение от противного. Формализуем данное свойство с помощью неравенств-ограничений 1), 2) (рис. 1c). Нули в уравнениях заменим на переменные и выпишем ограничения-неравенства 4) (рис. 1c).

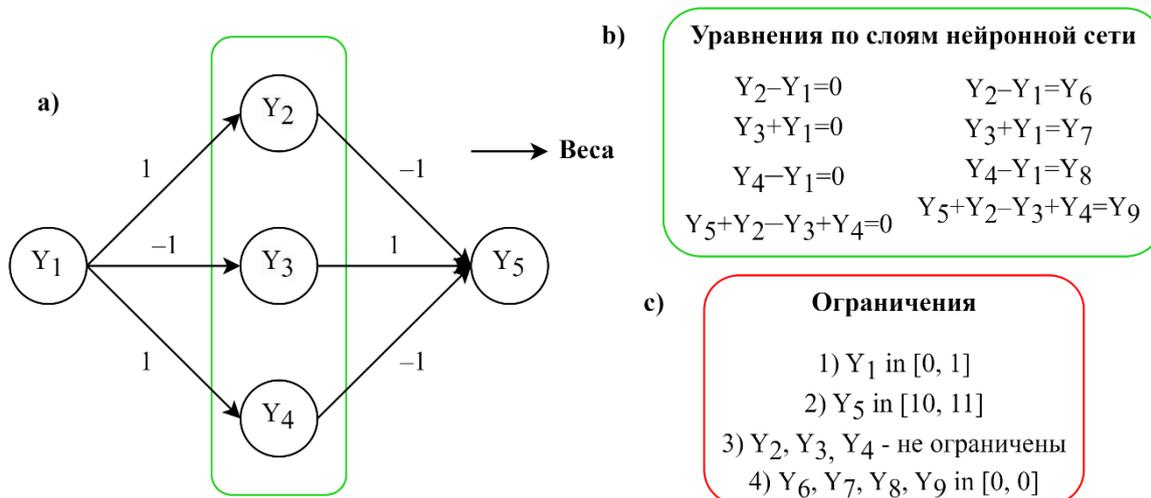


Рисунок 1. Однослойная нейросеть без функции активации. На рисунке а) изображен общий вид архитектуры; на б) выписаны линейные уравнения, соответствующие введённым переменным Y ; рисунок с) формализует заданные свойства ϕ в виде неравенств-ограничений

Теперь, когда выписана однородная система линейных уравнений и заданы ограничения в виде неравенств, можно воспользоваться алгоритмом для решения поставленной задачи. В качестве основы, например, можно взять алгоритм симплекс метода [9]. Такой подход подробнее описан в работе [7]. В результате работы симплекс метода будет дан ответ SAT, если система с ограничениями выполнимы и UNSAT – в противном случае, а также предоставлен контрпример, для которого система не выполняется.

Далее рассмотрим, как измениться постановка задачи в случае добавления функции активации, например ReLU.

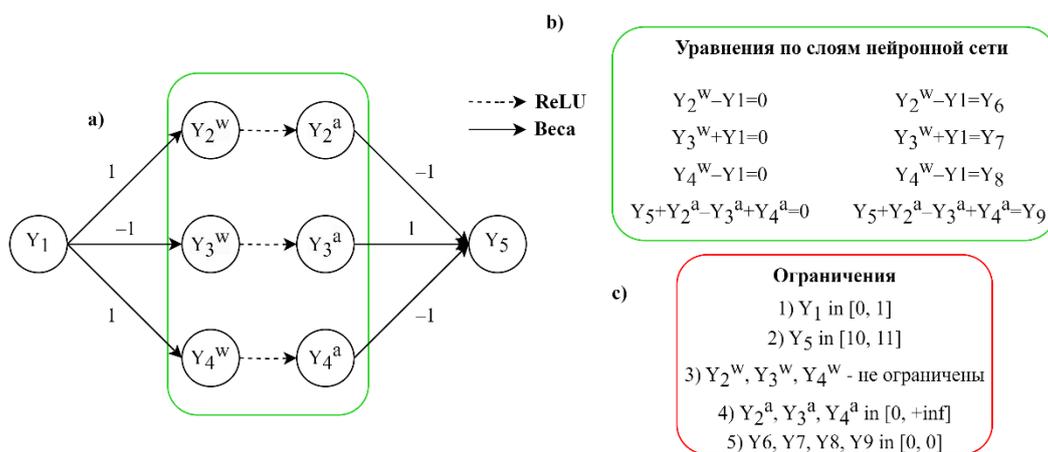


Рисунок 2. Однослойная нейросеть с функцией активации ReLU. На рисунке а) изображен общий вид архитектуры; на б) выписаны линейные уравнения, соответствующие введённым переменным Y ; рисунок с) формализует заданные свойства ϕ в виде неравенств-ограничений

Аналогично проделанному выше алгоритму выпишем систему уравнений с ограничениями (рис. 2b, рис. 2c). Поскольку функция активации ReLU совпадает с аргументом в случае его положительного значения, отразим это ограничение в неравенстве 4 (рис. 2c). После составления системы уравнений и ограничений работает алгоритм симплекс метода, и выдает SAT/UNSAT.

Рассмотренные выше методы применимы для проверки ограничений любых сетей с кусочно-линейными функциями и произвольным числом слоев. Алгоритмы основанные на доказательстве выполнимости свойств при заданных ограничениях производят полную верификацию (*complete verification*) и строгое математическое доказательство.

Методы, основанные на абстракции. Абстрактные методы (Abstraction-Based Verification) в отличие от предыдущего набора методов, основанных на ограничениях, выполняют *частичную верификацию*. Рассматриваемая в этом разделе группа методов основана на абстрактной интерпретации [6], хорошо изученном подходе для анализа программ.

Сначала мы задаем абстрактное множество возможных значений, например в виде интервала и затем продвигаем его вдоль распространения слоев, согласно их функциям. При линейных слоях перемножаются границы интервалов с заменой знака для отрицательных значений. В местах соединения нескольких нейронов значения соответствующих границ суммируются, тем самым расширяя область значений.

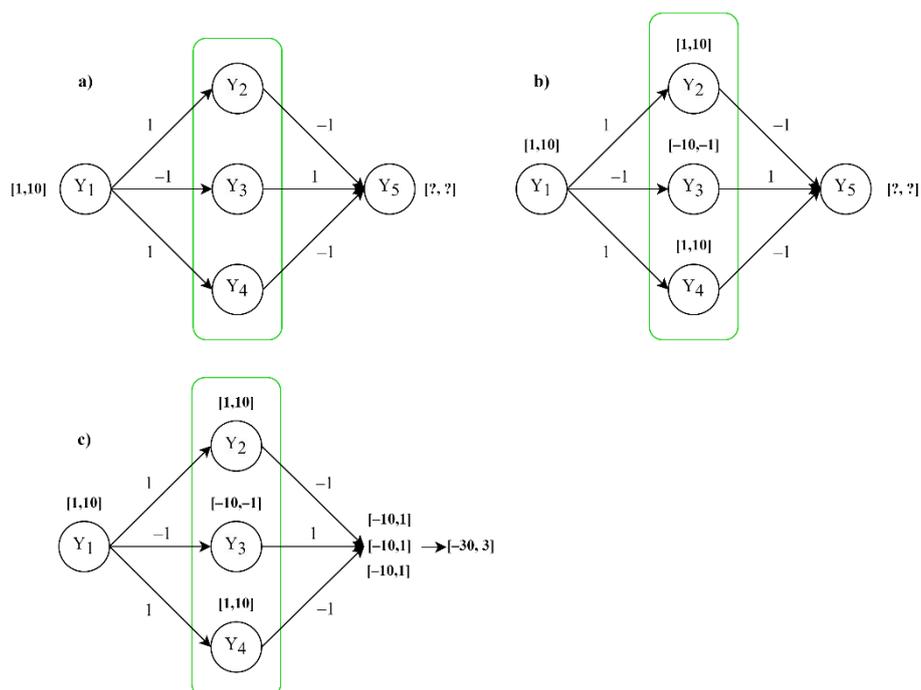


Рисунок 3. В качестве абстрактного множества выступают интервалы, последовательные этапы проброса интервалов (*interval bound propagation*)

Таким образом, мы получаем оценку снизу и сверху для выходного значения при заданном входном интервале.

3. Обзор программных систем верификации нейросетевых моделей

Несмотря на важность верификации нейросетевых алгоритмов, анализа их стабильности, на сегодняшний день не существует единой общей для всех классов нейросетей системы, который бы эффективно позволял проверять заданные свойства для моделей.

Внимания в этой области заслуживает проводимое ежегодно соревнование по верификации нейронных сетей (VNN-COMP [10]). Основной целью является формирование сообщества разработчиков, заинтересованных в создании перспективных программных средств проверки нейросетевых моделей. Аккуратная формулировка свойств для проверки сетей является нетривиальной задачей. Чтобы можно было обмениваться опытом верификации моделей организаторами были предложены форматы записи параметров сетей (ONNX, VNN-LIB) и свойств.

Нейросети представленные в этих форматах в рамках соревнования проверяются системами верификации, реализующими различные методы, запрограммированными на разных языках. В таблице 1 приведена сравнительная характеристика решений участников данного соревнования.

Таблица 1. Сравнительный анализ решений участников 4th International Verification of Neural Networks Competition (VNN-COMP 2023)

Название решения	Место / score	Поддержка GPU	Язык	Количество решенных задач	Время
α, β -CROWN	1 / 930.9	+	Python	10	5.7
FastBATLLNN	7 / 100.0	-	Python	1	0.3
Marabou	2 / 594.1	-	C++	9	1.1
NeuralSAT	4 / 547.0	+	Python	8	3.6
nenum	5 / 441.9	-	Python	6	0.9

NNV	6 / 176.4	-	MATLAB	6	15.9
PyRAT	3 / 585.5	+	Python	8	3.4

Стоит отметить что метод α, β -CROWN основан на абстрактной интерпретации, когда как подход Marabou использует верификацию на ограничениях при помощи модифицированного симплекс метода.

4. Адаптивная фильтрация: активное шумоподавление

В этом разделе на примере создания адаптивного фильтра, рассмотрим, как обеспечивается надежность и устойчивость подобных подходов. Для задачи активного шумоподавления рассмотрим классический вариант решения при помощи фильтра LMS: Least Mean Square Algorithm, а также предложенное авторами в работе [1] гибридное решение с нейросетевой моделью для прогноза данного фильтра.

Рассмотрим адаптивный фильтр LMS (Least Mean Square Algorithm) [11]. Обозначим $w(n)$ – веса фильтра, $y(n)$ – управляющий сигнал, $x(n)$ – входной сигнал, $d(n)$ – внешний сигнал (под который необходимо подстроиться). Схема общего фильтра представлена на рисунке 4.

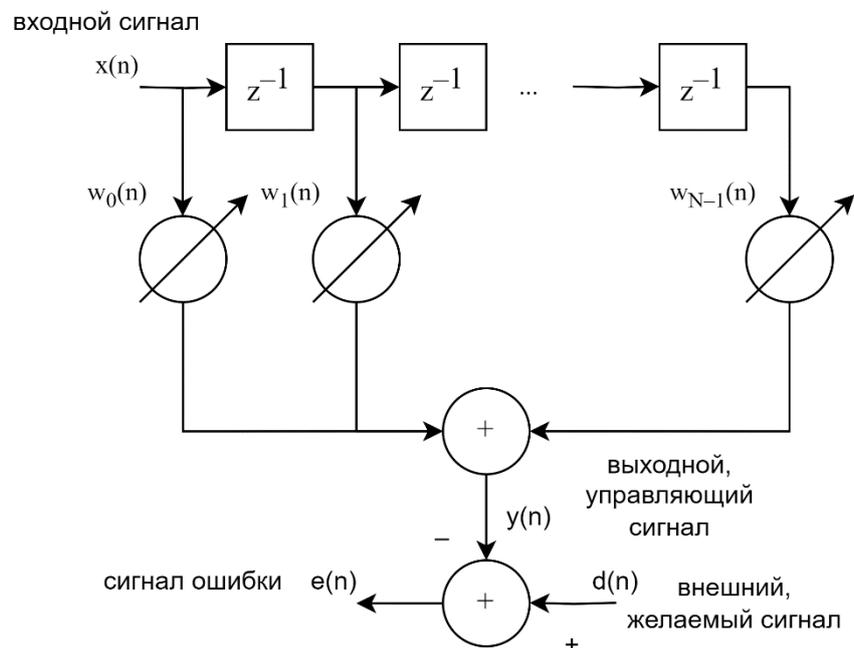


Рисунок 4. Общая схема адаптивного фильтра LMS

Формально задача состоит в минимизации $Ee^2(n)$, которая приводит к правилу обновления весов

$$w(n+1) = w(n) - \mu \nabla e^2(n) \quad (1)$$

Используя соотношения о сигнале ошибки, можно переписать (1) как

$$w(n + 1) = w(n) + 2\mu e(n)x(n) \quad (2)$$

Таким образом, для старта фильтрации необходимо проинициализировать вектор весов $w(0)$ и задать шаг μ . Получается, что на стабильность алгоритма влияет шаг, и начальный вектор весов. Поскольку мы обычно не знаем вид сигнала, веса инициализируются нулевым вектором, а для шага можно получить аналитическую оценку. Можно показать [11], что алгоритм LMS будет сходиться к весам оптимального фильтра (полученного аналитически из уравнения Винера — Хопфа), когда

$$0 < \mu < \frac{1}{\lambda_{max}} \quad (3)$$

где λ_{max} – собственное значение автокорреляционной матрицы $R = E[x(n)x^T(n)]$. Таким образом, гарантируется сходимость. Данное условие не является единственно возможным, но представляет собой доказательство сходимости для адаптивного фильтра, подобного рода гарантии нам хотелось бы иметь и для нейросетевых моделей.

Рассмотрим задачу активного шумоподавления, в которой будет работать адаптивный фильтр FxLMS (модификация LMS [12] для акустической среды) и параллельно нейросеть будет анализировать входной сигнал $x(n)$ и по нему прогнозировать коэффициенты фильтра, тем самым обновляя их при изменении внешнего шума. Рассматривается модель, которая предсказывает 15 возможных фильтров для различных типов сигналов (дорожный шум, белый шум, авиационный и другие). Подробное описание предлагаемой системы показано на рисунке 5.

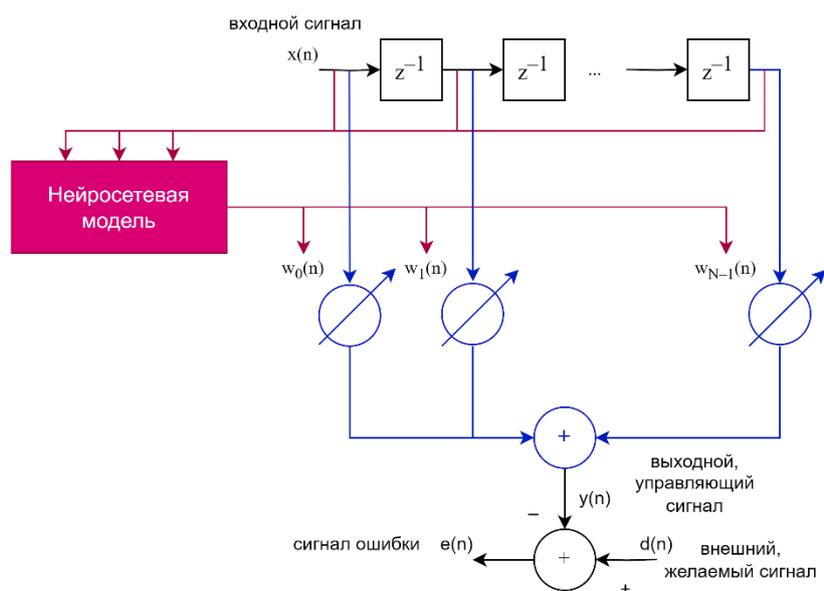


Рисунок 5. Схема работы адаптивного фильтра с нейросетевой моделью, предсказывающей его коэффициенты по входному сигналу $x(n)$

Для модели, используемой в системе шумоподавления актуальной является проверка, например, следующих свойств:

- Если магнитуа окружающего шума превышает определенный уровень шума, то веса предлагаемого нейросетью фильтра не могут быть нулевыми (тривиальное).

$$\exists n^*: \forall n > n^*, L(x(n)) > 40 \text{ dB} \Rightarrow \omega(n) \neq 0 \quad (4)$$

- Если распределение сигнала на входе отличается от равномерного (белый шум), то коэффициенты не могут совпадать с коэффициентами для подавления белого шума.

$$\exists n^*: \forall n > n^*, x(n) \sim U(-1,1) \Rightarrow |w(n) - w_{white}| < \varepsilon \quad (5)$$

Доказательство подобных свойств представляется необходимым перед внедрением сети в практическое использование.

5. Эксперименты

Рассмотрим веса упрощенной версии для нейросетевой модели, которая заменяет собой адаптивный фильтр, обученной на синтетических данных.

Модель представляет собой 3 линейных слоя с входом и выходом равным 320 семплов, что равняется 20 миллисекундам при частоте дискретизации равной 16kHz. Данная частота является популярной для обработки звуковых данных, в частности речевых данных – из-за стационарности высоты тона речевого сигнала порядка 20 миллисекунд.

В качестве эксперимента варьировалась размерность скрытого слоя с 512 до 1024 элементов, таким образом суммарное число параметров составило 377.920 и 1.017.664 соответственно.

В данной работе предложено программное решение, в котором предварительная подготовка данных (перевод параметров модели формата ONNX в ограничения), реализована в виде модуля на языке Python, а задача удовлетворения ограничений решается средствами языка Prolog (swi-prolog) с использованием библиотеки clpq [14].

Для сравнения была рассмотрено доказательство свойств той же модели с помощью системы Marabou [13]. Подробные результаты представлены в таблице 2.

Таким образом, мы видим, что оба подхода эффективно справились с предлагаемыми свойствами и ни одна из проверок не зациклилась. Мы видим также стабильное время работы реализации Marabou на C++. За счет использования Prolog и понижения точности удалось существенно сократить время проверки свойств по сравнению с реализацией Marabou. Также стоит отметить, что предложенная реализация использует существенно меньше памяти при проверке UNSAT свойства.

Таблица 2. Результаты экспериментов верификации четырех свойств для задачи фильтрации сигнала при помощи нейросетевой модели

Filter, frame=20ms		Предлагаемый подход			Marabou	
Св- во	Размер модели	Результат	Время, с	Память	Результат	Время, с
1	377.920	SAT	1.3s	170Mb	SAT	30.8s
2	377.920	SAT	9.7s	49Mb	SAT	29.2s
3	377.920	SAT	7.6s	216Mb	SAT	29.3s
4	377.920	UNSAT	7.7s	7.5Mb	UNSAT	31.6s
1	1.017.664	SAT	3.9s	487Mb	SAT	3m55s
2	1.017.664	SAT	62.5s	963Mb	SAT	4m18s
3	1.017.664	SAT	53.3s	643Mb	SAT	4m13s
4	1.017.664	UNSAT	56.5s	14.3Mb	UNSAT	4m14s

6. Заключение

В работе был проведен обзор существующих методов верификации нейросетевых моделей и предложен свой подход к решению задачи. Его исследование проводилось на тестовой нейросетевой модели шумоподавления. Построен программный стенд, на котором были верифицированы свойства нейросети в сравнении с системой Marabou. Сделаны вывод по применимости методов и даны оценки по времени и по памяти для рассматриваемых моделей.

Дальнейшее исследование связано с обобщением проверки свойств на новые типы сигналов и расширение класса нейросетевых моделей, с которыми работает верификатор, в частности моделей активного шумоподавления.

Литература

1. Полевой А.В., Корухова, Ю.С. Исследование алгоритмов активного шумоподавления для нестационарных сигналов // Тезисы докладов научной конференции "Тихоновские чтения" — М.: ООО "МАКС Пресс", 2022. — С. 51. — https://cs.msu.ru/sites/cmc/files/attachs/tezisytch2022_v_pechat.pdf
2. Fingscheidt T., Gottschalk H., Houben S. Deep neural networks and data for automated driving: Robustness, uncertainty quantification, and insights towards safety. — Springer Nature, 2022. — С. 427 — <https://doi.org/10.1007/978-3-031-01233-4>
3. Wu B., Wan A., Iandola F., Jin P. H., Keutzer K. SqueezeDet: Unified, Small, Low Power Fully Convolutional Neural Networks for Real-Time

- Object Detection for Autonomous Driving // IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW, Honolulu, HI, USA, 2017). — USA, 2017. — C. 446-454. — <https://doi.org/10.1109/CVPRW.2017.60>
4. Esteva A., Robicquet A., Ramsundar B. et al. A guide to deep learning in healthcare // Nature medicine. — 2019. — T. 25. — №. 1. — C. 24-29. — <https://doi.org/10.1038/s41591-018-0316-z>
 5. Owen M.P., Panken A., Moss R., Alvarez L. and Leeper C. ACAS Xu: Integrated collision avoidance and detect and avoid capability for UAS // 2019 IEEE/AIAA 38th Digital Avionics Systems Conference (DASC). — IEEE, 2019. — C. 1-10. — <https://doi.org/10.1109/DASC43569.2019.9081758>
 6. Cousot P. and Cousot R. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. // Proceedings of the 4th ACV SIGACT-SIGPLAN on Principles of Programming Languages (Los Angeles, California, USA, 1977). — ACM, 1977. — C. 238-252. — <https://doi.org/10.1145/512950.512973>
 7. Katz G., Barrett C., Dill D.L., Julian K., Kochenderfer M.J. Reluplex: An efficient SMT solver for verifying deep neural networks // Computer Aided Verification: 29th International Conference, CAV 2017, Heidelberg, Germany, July 24-28, 2017, Proceedings, Part I 30. — Springer International Publishing, 2017. — C. 97-117. — https://doi.org/10.1007/978-3-319-63387-9_5
 8. Ribeiro M. T., Singh S., Guestrin C. " Why should i trust you?": Explaining the predictions of any classifier // Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. — ACM, 2016. — C. 1135-1144. — <https://doi.org/10.1145/2939672.293977>
 9. Nelder J. A., Mead R. A simplex method for function minimization // The computer journal. — 1965. — T. 7. — No. 4. — C. 308-313. — <http://dx.doi.org/10.1093/comjnl/7.4.308>
 10. Brix C. et al. The fourth international verification of neural networks competition (vnn-comp 2023): Summary and results // arXiv preprint arXiv:2312.16760. — 2023
 11. Farhang-Boroujeny B. Adaptive filters: theory and applications. — UK: John Wiley & Sons, 2013. — C. 778
 12. Kim I.-S., Na H.-S., Kim K.-J., Park Y. Constraint filtered - x and filtered - u least - mean - square algorithms for the active control of noise in ducts // The Journal of the Acoustical Society of America. — 1994. — T. 95. — No. 6. — C. 3379-3389. — <https://doi.org/10.1121/1.409957>
 13. Katz G. et al. The Marabou framework for verification and analysis of deep neural networks // Computer Aided Verification: 31st International Conference, CAV 2019, New York City, NY, USA, July 15-18, 2019,

Proceedings, Part I 31. – Springer International Publishing, 2019. – C. 443-452 — https://doi.org/10.1007/978-3-030-25540-4_26

14. SWI-Prolog manual. A.10 library(clpqr): Constraint Logic Programming over Rationals and Reals <https://www.swi-prolog.org/man/clpqr.html>

References

1. Polevoi A.V., Korukhova Y.S. Issledovanie algoritmov aktivnogo shumopodavlenija dlja nestacionarnyh signalov // Tezisy dokladov nauchnoj konferencii "Tihonovskie chtenija" — M.: OOO "MAKS Press". — P. 51. — https://cs.msu.ru/sites/cmc/files/attachs/tezisytch2022_v_pechat.pdf
2. Fingscheidt T., Gottschalk H., Houben S. Deep neural networks and data for automated driving: Robustness, uncertainty quantification, and insights towards safety. — Springer Nature, 2022. — P. 427 — <https://doi.org/10.1007/978-3-031-01233-4>
3. Wu B., Wan A., Iandola F., Jin P. H., Keutzer K. SqueezeDet: Unified, Small, Low Power Fully Convolutional Neural Networks for Real-Time Object Detection for Autonomous Driving // IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW, Honolulu, HI, USA, 2017). — USA, 2017. — P. 446-454. — <https://doi.org/10.1109/CVPRW.2017.60>
4. Esteva A., Robicquet A., Ramsundar B. et al. A guide to deep learning in healthcare // Nature medicine. – 2019. – T. 25. – №. 1. – P. 24-29. — <https://doi.org/10.1038/s41591-018-0316-z>
5. Owen M.P., Panken A., Moss R., Alvarez L. and Leeper C. ACAS Xu: Integrated collision avoidance and detect and avoid capability for UAS // 2019 IEEE/AIAA 38th Digital Avionics Systems Conference (DASC). – IEEE, 2019. — P. 1-10. — <https://doi.org/10.1109/DASC43569.2019.9081758>
6. Cousot P. and Cousot R. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. // Proceedings of the 4th ACV SIGACT-SIGPLAN on Principles of Programming Languages (Los Angeles, California, USA, 1977). — ACM, 1977. — P. 238–252. — <https://doi.org/10.1145/512950.512973>
7. Katz G., Barrett C., Dill D.L., Julian K., Kochenderfer M.J. Reluplex: An efficient SMT solver for verifying deep neural networks // Computer Aided Verification: 29th International Conference, CAV 2017, Heidelberg, Germany, July 24-28, 2017, Proceedings, Part I 30. – Springer International Publishing, 2017. – P. 97-117. — https://doi.org/10.1007/978-3-319-63387-9_5
8. Ribeiro M. T., Singh S., Guestrin C. " Why should i trust you?": Explaining the predictions of any classifier // Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. — ACM, 2016. — P. 1135-1144. — <https://doi.org/10.1145/2939672.293977>

9. Nelder J. A., Mead R. A simplex method for function minimization //The computer journal. — 1965. — T. 7. — No. 4. — P. 308-313. — <http://dx.doi.org/10.1093/comjnl/7.4.308>
10. Brix C. et al. The fourth international verification of neural networks competition (vnn-comp 2023): Summary and results //arXiv preprint arXiv:2312.16760. — 2023
11. Farhang-Boroujeny B. Adaptive filters: theory and applications. — UK: John Wiley & Sons, 2013. — P. 778
12. Kim I.-S., Na H.-S., Kim K.-J., Park Y. Constraint filtered - x and filtered - u least - mean - square algorithms for the active control of noise in ducts //The Journal of the Acoustical Society of America. — 1994. — T. 95. No. 6. — P. 3379-3389. — <https://doi.org/10.1121/1.409957>
13. Katz G. et al. The Marabou framework for verification and analysis of deep neural networks //Computer Aided Verification: 31st International Conference, CAV 2019, New York City, NY, USA, July 15-18, 2019, Proceedings, Part I 31. — Springer International Publishing, 2019. — P. 443-452 — https://doi.org/10.1007/978-3-030-25540-4_26
14. SWI-Prolog manual. A.10 library(clpqr): Constraint Logic Programming over Rationals and Reals <https://www.swi-prolog.org/man/clpqr.html>