Методы формальной верификации искусственных нейронных сетей: обзор существующих подходов

Е.Н. Строева, А.А. Тонких

***Аннотация*—Среди последних публикаций, содержащих обзор и систематизацию алгоритмов формальной верификации нейронных сетей, наиболее полный анализ предлагает классификацию алгоритмов с использованием трёх свойств: достижимости, оптимизации и поиска. Методы, основанные на достижимости, выбирают данные из области входных значений, исходя из заданных заранее на них ограничений, и аппроксимируют это множество при помощи символьных математических конструкций. Поскольку приближение происходит на каждом нейроне, накапливается «раздутие области значений». Соответственно, главной проблемой такого подхода является переаппроксимация, то есть слишком широкое выходное множество. Также большой проблемой является сохранение линейности после применения функции активации ReLU, нужное для применения алгоритма обратного распространения ошибки.**

В данной статье изучены и систематизированы математические конструкции и подходы к приближению точечных данных плоскими непрерывными множествами с наименьшей аппроскимацией из возможных. Также рассматриваются наиболее эффективные методы для разрешения проблемы применения функции активации ReLU на тех нейронах, где возникает неоднозначность.

***Ключевые слова*—формальная верификация, машинное обучение**

I. Введение

Применение нейронных сетей в критически важных областях, таких как беспилотное управление, детектирование опухолей, управление электростанциями становится всё более обширным, а ошибки в их работе имеют всё более опасные и дорогостоящие последствия. Возникает проблема подтверждения надёжности получаемых результатов — проблема формальной верификации контроллеров под управлением нейронных сетей.

Традиционно проверка нейронных сетей сосредоточена на тестировании — оценке сети на большом наборе точек во входном пространстве и определении того, соответствуют ли её выходы желаемым на этом наборе. Однако, невозможно проверить все возможные входные данные, поскольку входное пространство очень часто фактически бесконечно по мощности.

Формальная верификация обеспечивает ортогональную альтернативу тестированию, она предоставляет математические гарантии в отношении верной работы искусственной нейронной сети для любого входа из пространства возможных входных данных.

В данной работе представлено исследование методов формальной верификации искусственных нейронных сетей. В процессе исследования формулируется весь необходимый математический аппарат для формального описания темы, а также выделяются наиболее действенные и стабильные алгоритмы формальной верификации нейронных сетей. Углубляясь в исследование вопроса, более подробно рассматриваются математические методы, лежащие в основе выделенных алгоритмов формальной верификации, основанных на свойстве достижимости каждого нейрона, а также проводится систематизация выбранных методов.

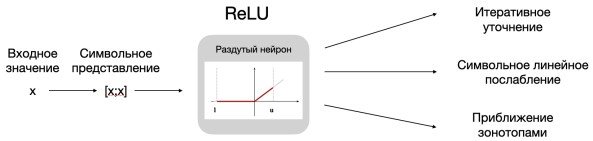


Рис. 1. Схема изложенных в данной статье методов

II. Основные подходы к формальной верификации

Существует четыре основных подхода к верификации искусственных нейронных сетей [1]:

A. Решение SAT/SMT

Нейронная сеть и система введённых ограничений представляются как булева формула и могут быть выражены через конъюнктивную нормальную форму [2], [3]. Далее, созданная формула проверяется автоматическим решателем SAT. SMT — вариант SAT для k-значной логики. Наличие выходных данных SAT означает, что найдено отрицание свойства — контрпример. UNSAT означает, что нет контрпримера, другими словами — свойство выполняется.

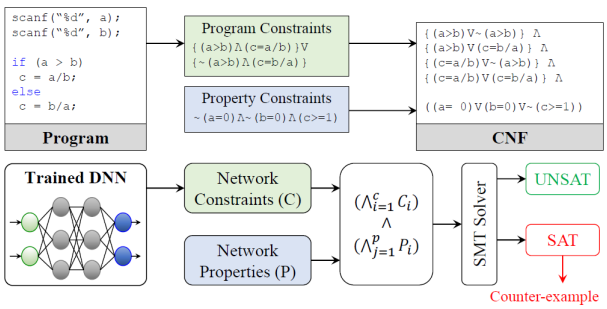


Рис. 2. SMT подход в формальной верификации ИНС [4]

B. Линейное программирование

При использовании линейного программирования (linear programming, LP) система введённых ограничений определяется как набор линейных ограничений (система линейных уравнений и неравенств), а свойство как целевая функция, которую надо максимизировать или минимизировать в зависимости от условия задачи [5], [6]. Проверка происходит автоматически при помощи линейных программаторов.

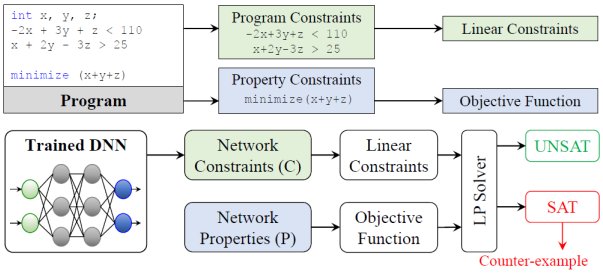


Рис. 3. Линейное программирование в формальной верификации ИНС [4]

C. Доказательство теорем

Система введённых ограничений представляется в виде модели, управляемой математическими принципами, свойство — цель доказательства [7]. Принцип подхода: использовать аксиомы и правила чтобы проверить, соответствуют ли свойства созданной модели исходной системе.

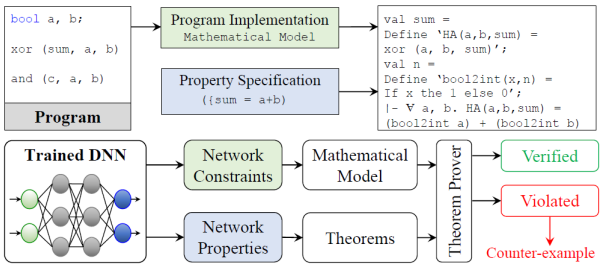


Рис. 4. Доказательство теорем в формальной верификации ИНС [4]

D. Неполная проверка

Данный метод использует в своей основе создание упрощённой модели заданной системы, обладающей теми же свойствами [8]. Полученная модель не является точным представлением реальной системы, а является чрезмерным приближением. Выполняется проверка созданной модели с использованием других подходов.

III. Применяемые математические методы

Наиболее часто встречающиеся методы верификации нейронных сетей основаны на применении SAT/SMT-решателей и LP-решателей, поэтому основная задача при создании верификационного алгоритма — перевести заданные ограничения в вид, доступный для работы решателя. Точнее, для SAT/SMT-решателей перевести заданные в выбранной задаче ограничения, как правило представленные отрезком, в символьные представления,

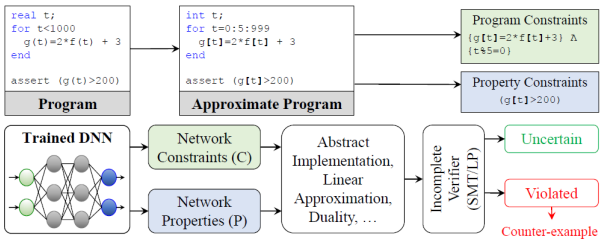


Рис. 5. Неполная проверка в формальной верификации ИНС [4]

связанные конъюнкциями или дизъюнкциями, а для LP-решателей — в систему линейных неравенств. Сама нейронная сеть тогда представляется как модель, сохраняющая все свойства нейронной сети, но работающая со специально введёнными символьными представлениями, различными для каждого из подходов.

Наиболее частым подходом к верификации является комбинация неполной проверки и использования SAT/SMT-решателей или неполной проверки и LP-решателей. В общем случае задача верификации нейронной сети — проверить её работу на всём возможном множестве входных данных. Входные данные — это векторы, точки в многомерном пространстве, однако, в данной задаче нужно мыслить иначе — «расширить» точки на всё возможное пространство входов, которое представляет собой некоторую область, и оценить размер выходных данных, которые может выдавать нейронная сеть после применения к любому элементу из этой области.

Другими словами, сложность состоит в том, чтобы перевести задачу из дискретной в непрерывную с сохранением всех свойств нейронной сети и однозначным соответствием входного и выходного множеств. Однако, при расширении области входных данных и последующем применении нейронной сети к ним, границы выходного множества также расширяются, и ставится задача придумать методы, позволяющие сузить границы выходного множества, сохранив при этом все взаимосвязи и прослеживаемость переходов между нейронами.

Ключевая идея большинства применяемых методов — перейти от векторов к функциям, причём сделать это на каждом шаге: оценить входное и выходное множество для каждого нейрона, а также измерить результаты применения функции активации. Одним из самых эффективных методов в этой области является интервальный анализ и его аналоги [9].

A. Символьный интервальный анализ

Символьный интервальный анализ [10] содержит в своей основе следующую идею — замена каждого входного интервала на переменную (символ), далее производится один шаг по нейронной сети и получаются символьные интервалы (интервалы в которых есть символы, переменные), затем итерационный процесс повторяется. Идея применяется во всех достижимых алгоритмах, служит основой алгоритма ReLUVal [10].

Рассмотрим различные подходы к оценке интервалов: предположим, у нас есть два самолёта и нейронная сеть, которая оценивает поведение одного из них, используя расстояние до другого и угол возможного поворота.

Входные данные: расстояние от второго самолёта $\in [4;6]$, возможный угол поворота второго самолёта $\in [1;5]$. Выходные данные: безопасный угол поворота нашего самолёта должен быть меньше 20, то есть $\in [0;20]$.

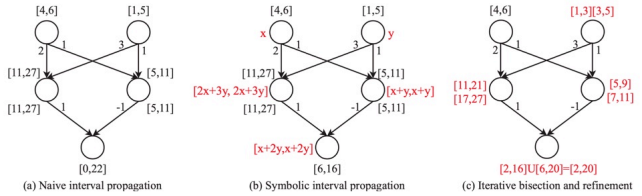


Рис. 6. Символьный интервальный анализ и итеративное уточнение [10]

Как хорошо видно из рисунка 6, общепринятый математический подход к работе с интервалами — (a) Naive interval propagation: левая граница интервала складывается из преобразований левых границ каждого интервала, правая граница из соответствующих преобразований правых границ — приводит к результатам, лежащим вне допустимых границ выходного множества.

Рассмотрим символьный подход (b) Symbolic interval propagation, основанный на замене интервалов переменными-символами. Функция активации нейронной сети применяется уже не к числовым значениям, а к символам. На каждом шаге производятся линейные преобразования уже над символами. На выходе получаются линейные комбинации входных интервалов в символьном виде, подставляются исходные числовые границы интервалов, и получается интервал, удовлетворяющий заданным ограничениям.

Математическое обоснование:

Пусть x — действительное значение. Его можно представить вырожденным интервалом $[x;x]$.

Интервальным представлением функции $f(x)$ называется новая функция F, такая что при применении функции F к интервалу $[x;x]$ получается такой же результат, как и при применении функции f к значению x:

$$F([x;x]) = f(x) \qquad \forall x \in X.$$

Другими словами, появилась возможность перейти к символьной арифметике, применять функции к вырожденным интервалам вида $[x;x]$ и к стандарнтым интервалам вида $[x;y]$.

Рассмотрим на примере (b): выбирается некоторое число x из интервала $X = [4;6]$, оно представляется в качестве интервала $[x;x]$ с двумя равными границами. Аналогично для интервала $Y = [1;5]$ элемент y представляется как $[y;y]$. Функция-нейронная сеть f, которая работает с элементами x и y заменяется на F, которая работает с интервалами $[x;x]$ и $[y;y]$, на каждом шаге умножая границы на заданные веса и получая снова интервалы подобного вида: $[2x+3y;2x+3y]$, $[x+y;x+y]$.

На выходе получается интервал $[x+2y;x+2y]$, а с учётом того, что $x \in [4;6]$ и $y \in [1;5]$, применяется обычный метод работы с интервалами: для левой границы $x = 4, y = 1$, результат 6, для правой границы $x = 6, y = 5$, результат 16. Итого выходное множество удовлетворяет заданным ограничениям $[6;16] \subseteq [0;20]$.

В общем виде, помимо умножения на веса на каждом шаге нейронной сети, происходит применение функции активации, а во всех основных работах по формальной верификации искусственных нейронных сетей используется ReLU.

Пусть $z' = Eq = [l;u]$ — символьное представление интервала, получившегося после умножения на веса на некотором промежуточном нейроне. Тогда после применения функции активации ReLU для получения выходного значения $z = ReLU(z')$ может получиться несколько возможных результатов:

$z = [Eq;Eq]$, если $l \geq 0$ — все значения из интервала положительные, сам интервал остался неизменным, нейрон активирован;

$z = [0;0]$, если $u \leq 0$ — все значения интервала отрицательные, выходное значение 0, нейрон не активирован;

$z = [l;u]$, если $l < 0 < u$ — интервал невозможно перевести функцией ReLU в интервал, необходимо дальнейшее исследование.

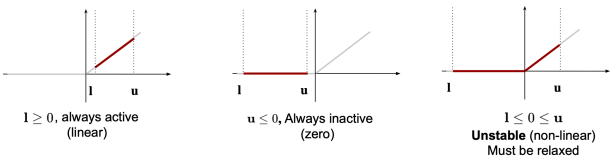


Рис. 7. Варианты расположения символьного интервала после применения функции активации ReLU

Проблема возникает в третьем случае, когда границы интервала расположены по разные стороны от нуля. Напомним, что результат применения функции ReLU всегда неотрицательный. Нейроны 3 типа будем называть «раздутыми», а основная задача алгоритма верификации как раз заключается в том, чтобы понять, как обработать результат и в каком виде передать его на следующий нейрон. В этом случае границы интервала $[l;u]$ называют пре-активационными, так как они не дают точного ответа, будет активирован данный нейрон или нет.

Сконцентрируем внимание читателя на основной проблеме достижимости алгоритма формальной верификации нейронных сетей. После применения функции активации ReLU нужно получить результат, в таком виде, чтобы можно было передать его на следующий нейрон. Этот результат должен быть выражен через два «объекта» (числа или символьно-заданные) — концы отрезка $[l;u]$, которые мы подаём на вход функции активации ReLU рассматриваемого нейрона. Так как у нас алгоритмы должны соответствовать принципу достижимости каждого нейрона, надо понимать, что происходит на каждом нейроне, что приходит на вход, и что получается на выходе после применения функции ReLU. Идея сохранить как можно больше информации, но при этом не раздуть полученное значение переаппроксимацией.

Итак, есть 3 параметра: входное значение z, которое записано как символ, границы оценивающего его отрезка l и u, и наша задача состоит в том, чтобы сконструировать при помощи них какое-то «разумное» множество, которое можно потом передать на следующий нейрон. Чтобы умножить на веса это множество, дальше снова применять функцию ReLU и получать осмысленный результат.

B. Итеративное уточнение

Наиболее простая идея для работы с раздутыми нейронами — сохранение максимально возможного разброса, и замена одной из границ нулевым значением. Вторая составляющая алгоритма формальной верификации ReLUVal [10].

Математическое обоснование:

Пусть есть нейрон A на первом слое, где $x_1, \dots, x_d$ — входные данные, а $w_1, \dots, w_d$ — соответствующие веса. Тогда границы символьного интервала будут одинаковы и равны:

$$Eq^A_{up}(X) = Eq^A_{low}(X) = w_1x_1 + \dots + w_dx_d.$$

Если нейрон A лежит на некотором промежуточном слое, то встаёт задача сохранить максимальный разброс — наибольшую из возможных длин интервала. Определим как W_+ и W_- — положительные и отрицательные веса на текущем слое, $Eq^{A_{prev}}_{low}$ и $Eq^{A_{prev}}_{up}$ — границы интервала, пришедшие с предыдущего слоя. Тогда для нижних и верхних границ получается оценка:

$$Eq^A_{low}(X) = W_+Eq^{A_{prev}}_{low}(X) + W_-Eq^{A_{prev}}_{up}(X) =$$
$$= [w_-l; w_-u] = [\underline{Eq_{low}(X)}; \overline{Eq_{low}(X)}]$$
$$Eq^A_{up}(X) = W_+Eq^{A_{prev}}_{up}(X) + W_-Eq^{A_{prev}}_{low}(X) =$$
$$= [w_+l; w_+u] = [\underline{Eq_{up}(X)}; \overline{Eq_{up}(X)}]$$

Получается, что для максимальной величины длины интервала имеет смысл взять интервал $[\underline{Eq_{low}(X)}; \overline{Eq_{up}(X)}]$, и далее возможны следующие варианты:

1) если $\underline{Eq_{low}(X)} > 0$, просто берётся указанный интервал и производится переход на следующий слой;
2) если $\underline{Eq_{low}(X)} < 0$, нижняя граница определяется как 0 — берётся интервал $[0; \overline{Eq_{up}(X)}]$ и отдельно рассматривается верхняя граница;
3) если $\overline{Eq_{up}(X)} > 0$, то производится переход к следующему слою по интервалу из предыдущего пункта;
4) если $\overline{Eq_{up}(X)} < 0$, то интервал определяется как $[0; 0]$ и производится переход к следующему слою.

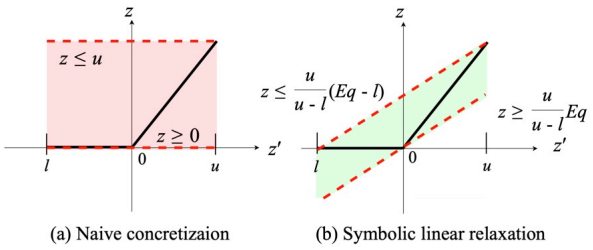
C. Символьное линейное послабление

Символьное линейное послабление [11] подразумевает приближении функции активации ReLU плоским многоугольником, задание его при помощи линейных неравенств для символов, и запись результата применения снова будет в символьном виде. По сути это расширение символьного интервального анализа на систему линейных ограничений, вместо одного интервала.

Математическое обоснование:

При обычном подходе мы заключаем результат применения функции активации в горизонтальную полосу, но это не самый оптимальный вариант с точки зрения эффективности аппроксимации. Более эффективный подход — заключить в параллелограмм:

С помощью метода 1 уже получилось определить нижние и верхние границы для интервала, полученного на предыдущем слое: $Eq_{low} = [l_{low}; u_{low}]$ и $Eq_{up} = [l_{up}; u_{up}]$, тогда после применения функции активации ReLU получим:



(a) Naive concretizaion (b) Symbolic linear relaxation

Рис. 8. Пример символьной линейной релаксации — приближение параллелограммом [11]

$$ReLU(Eq_{low}) = \frac{u_{low}}{u_{low} - l_{low}}(Eq_{low})$$

$$ReLU(Eq_{up}) = \frac{u_{up}}{u_{up} - l_{up}}(Eq_{up} - l_{up})$$

Другими словами, вместо интервавала берётся многоугольник, который представляется символом, к нему применяется функция ReLU, результатом которой будет снова некоторый многогранник, который снова можно представить символом. Следующие итерации аналогичны.

Резюмируя идею применения символьных методов к интервалам и функциям активации, можно заметить, что здесь прослеживается связь с идеями геометрической вероятности. Задача состоит в том, чтобы построить непрерывное множество, содержащее все входные данные, но с минимально возможной площадью.

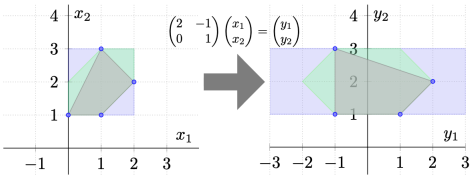


Рис. 9. Box, polyhedra и zonotope — сиреневый, серый и зелёный соответственно [12]

На рисунке 9 показаны голубые точки — это входные данные. Самым точным приближением будет многоугольник серого цвета box, однако с ним сложно работать. Приближение с помощью polyhedra (сиреневый многоугольник) имеет довольно сложное для последующего анализа описание в виде линейных ограничений. Именно поэтому был придуман и получил широкое применение многоугольник zonotope, представленный на рисунке зелёным цветом. Следует заметить, что все указанные многоугольники при действии аффинных функций переходят в многоугольники такого же вида. В общем случае многогранники задаются следующими системами неравенств:

Box (серого цвета): $a \le x_i \le b, a, b \in \mathbb{R}, a \le b$

Polyhedra (розового цвета): $A\bar{x} \le \bar{b}$ — линейные ограничения.

Zonotope (зелёного цвета) — более сложная в представлении структура, будет рассматриваться подробнее далее.

D. Абстракция с применением зонотопов

Абстракция с применением зонотопов [12], [13], [14], [15] происходит из идеи задать приближающий многоугольник в интервальной аффинной форме, что позволит также построить приближения нелинейных функций активации «типа S» — сигмоиду и гиперболический тангенс. Применяется в алгоритмах AI^2 [12] и DeepZ [14].

Математическое обоснование:

Основная проблема символьного подхода к интервалам заключается в том, что этот подход просто ограничивает значения по каждому измерению, но не связывает значения различных измерений. Как решение этой проблемы была придумана концепция зонотопа — n-мерного многоугольника, который задаётся своим центром, а каждая его точка определяется «перемещением» центра вдоль заданных направлений. То есть зонотоп — не статическая, а динамическая фигура, границы которой изменяются в совокупности, в зависимости от ограничений по всем направлениями.

$$\left\{ \left(c_{10} + \sum_{i=1}^{m} c_{1i} \cdot \epsilon_i, \ldots, c_{n0} + \sum_{i=1}^{m} c_{ni} \cdot \epsilon_i \right) \middle| \right.$$

$$\left. \epsilon_i \in [-1;1], c_{j_i} \in \mathbb{R}, i = \overline{1,m}, j = \overline{1,n} \right\}$$

В литературе также можно встретить следующие обозначения:
$z(\bar{\epsilon}) = M\bar{\epsilon} + \bar{b}$ или $z : [a_1;b_1] \times \cdots \times [a_m;b_m] \to \mathbb{R}^n$ — центр определяется bias-вектором $\bar{b}$, M — матрица, определяющая границы zonotope вокруг $\bar{b}$, $\bar{\epsilon}_i \in [a_i;b_i]$, где $i = \overline{1,m}$.

Рассмотрим примеры простейших зонотопов [15] для лучшего понимания концепции «динамического многоугольника»:

Однонаправленный зонотоп с 1 порождающей переменной:

$$\{c_0 + c_1\epsilon_1 | \epsilon_1 \in [-1;1]\}$$

что то же самое, что и интервал $[c_0 - |c_1|; c_0 + |c_1|]$ с центром в точке c_0.

Двунаправленный зонотоп с 1 порождающей переменной:

$$\{(2 + \epsilon_1; 2 + \epsilon_1) | \epsilon_1 \in [-1;1]\}$$

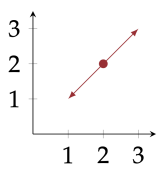


Рис. 10. Двунаправленный зонотоп с одной порождающей переменной [15]

Поскольку одна и та же переменная одинаково меняет оба направления, то центральная точка перемещается сразу по двум направлениям, что даёт наклонный отрезок.

Двунаправленный зонотоп с 2 порождающими переменными может иметь несколько интерпретаций:

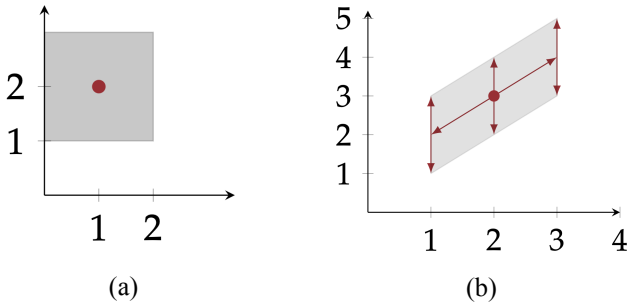


Рис. 11. Примеры двунаправленных зонотопов с двумя порождающими переменными [15]

Формульное описание зонотопа (a):

$$\{(1 + \epsilon_1; 2 + \epsilon_2) | \epsilon_1, \epsilon_2 \in [-1;1]\}$$

Формульное описание зонотопа (b):

$$\{(2 + \epsilon_1; 3 + \epsilon_1 + \epsilon_2) | \epsilon_1, \epsilon_2 \in [-1;1]\}$$

В случае а) получается, что центр перемещается вдоль горизонтальной оси по отрезку $[0;2]$, и вдоль вертикальной оси по отрезку $[1;3]$ «не синхронно».

В случае б) происходит «синхронное» перемещение центра вдоль наклонного отрезка с границами в точках $(1;2)$ и $(3;4)$ (как в случае с двунаправленным зонотопом с 1 порождающей переменной) благодаря ϵ_1, и этот же наклонный отрезок сдвигается вдоль горизонтали вверх и вниз на 1 благодаря ϵ_2.

В общем виде символьное интервальное представление зонотопа задаётся как:

$$\hat{x} = [\alpha_0;\beta_0] + \sum_{i=1}^{p} [\alpha_i;\beta_i] \cdot \epsilon_i,$$

где $\alpha_0, \beta_0, \alpha_i, \beta_i \in \mathbb{R}, \epsilon_i \in [a_i;b_i] \subseteq [0;1], i = \overline{1,p}$

Рассмотрим, как из двух зонотопов получить фигуру такого же вида [13]:

$$\hat{x} = \alpha_0^x + \sum_{i=1}^{n} \alpha_i^x \epsilon_i, \hat{y} = \alpha_0^y + \sum_{i=1}^{n} \alpha_i^y \epsilon_i, \hat{z} = \alpha_0^z + \sum_{i=1}^{n} \alpha_i^z \epsilon_i$$

Алгоритм объединения зонотопов $\hat{x}$ и $\hat{y}$, результатом которого снова является зонотоп $\hat{z}$ заключается в вычислении коэффициентов для зонотопа $\hat{z}$ [13]:

$\alpha_0^z = \text{mid}(\gamma(\hat{x}) \cup \gamma(\hat{y}))$ (центр зонотопа of $\hat{z}$)

$\alpha_i^z = \underset{\min(\alpha_i^x, \alpha_i^y) \le \alpha \le \max(\alpha_i^x, \alpha_i^y)}{\text{argmin}} (|\alpha|), \forall i \ge 1$
(коэффициент ϵ_i)

$\beta^z = \sup(\gamma(\hat{x}) \cup \gamma(\hat{y})) - \alpha_0^z - \sum_{i \ge 1} |\alpha_i^z|$
(коэффициент ϵ_U)

Здесь интервальное уточнение зонотопа задано функцией

$$\gamma(\hat{x}) = \left[\alpha_0^x - \sum_{i=1}^{n} |\alpha_i^x|, \alpha_0^x + \sum_{i=1}^{n} |\alpha_i^x| \right]$$

$\text{mid}([a,b]) := \frac{1}{2}(a+b)$

$\underset{a \le x \le b}{\text{argmin}}(|x|) := \{x \in [a,b], |x| \text{минимально}\}$

Наглядный пример применения данного алгоритма:

$$\begin{pmatrix} \hat{x} = 3 + \epsilon_1 + 2\epsilon_2 \\ \hat{u} = 0 + \epsilon_1 + \epsilon_2 \end{pmatrix} \cup \begin{pmatrix} \hat{y} = 1 - 2\epsilon_1 + \epsilon_2 \\ \hat{u} = 0 + \epsilon_1 + \epsilon_2 \end{pmatrix} =$$

$$= \begin{pmatrix} \hat{x} \cup \hat{y} = 2 + \epsilon_2 + 3\epsilon_U \\ \hat{u} \cup \hat{u} = 0 + \epsilon_1 + \epsilon_2 \end{pmatrix}$$

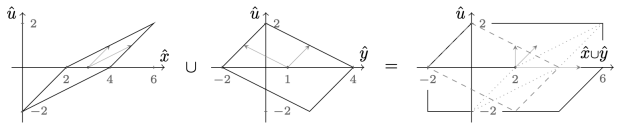


Рис. 12. Пример объединения двух зонотопов [13]

Отдельно отметим основное преимущество зонотопа — после объединения нескольких зонотопов в один, сохраняется его форма записи в виде линейной функции, что позволяет легко применять алгоритм обратного распространения ошибки, сохранять взаимосвязи междй нейронами и анализировать поведение нейронной сети.

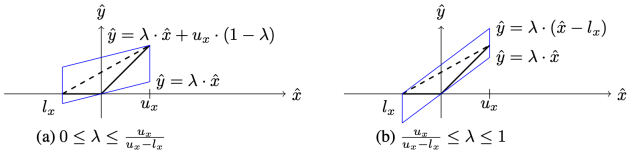


Рис. 13. Приближение ReLU с помощью polyhedra и zonotope [14]

На рисунке 13 представлено приближение функции ReLU на заданном отрезке при помощи параллелограммов, которые можно представить как в виде линейных ограничений с параметром λ, так и в с помощью зонотопа:

Разберём подробнее принцип построения границ параллелограмма. Нижней границей выбирается прямая, касающаяся функции ReLU в точке $(0; 0)$, заданная уравнением $\hat{y} = \lambda \cdot \hat{x}$. Параллельную ей верхнюю границу можно выбрать двумя способами: либо проходящей через правую крайнюю точку с абциссой u_x, подставляя эту точку в уравнение прямой получим коэффициент сдвига и уравнение $\hat{y} = \lambda \cdot \hat{x} + u_x \cdot (1 - \lambda)$, либо через крайнюю левую точку с абциссой l_x, соответственно получим уравнение $\hat{y} = \lambda \cdot (\hat{x} - l_x)$.

Новый параметр λ надо выразить через уже имеющиеся u_x и l_x. Для этого вводится условие минимума площадей заданных параллелограммов. Геометрически площадь считается как произведение стороны на проведённую к ней высоту, в обоих случаях в качестве высоты будет выступать сам отрезок $[l_x; u_x]$ длиной $(u_x - l_x)$, а в качестве основания — левая вертикальная граница, длина которой в случае (a) равна $\lambda \cdot u_x + u_x \cdot (1-\lambda) - \lambda \cdot u_x = u_x \cdot (1 - \lambda)$, в случае (b) — $\lambda \cdot (u_x - l_x) - \lambda \cdot u_x = -\lambda \cdot l_x$.

Таким образом площади указанных параллелограммов

$$S_a = (u_x - l_x) \cdot u_x \cdot (1 - \lambda), \qquad S_b = (u_x - l_x) \cdot (-\lambda) \cdot l_x$$

И после взятия производной получается, что обе площади принимают минимальные значения при $\lambda_{opt} = \frac{u_x}{u_x - l_x}$.

Для получения формулы зонотопа вводится новый параметр μ, отвечающий за центр параллелограмма, другими словами — отрезок прямой $\hat{y} = \lambda_{opt} \hat{x}$ поднимается вверх на половину длины вертикального отрезка, равного $-\lambda_{opt} \cdot l_x$, и будет ездить вверх и вниз отсюда $\mu_{opt} = \hat{y}_{opt} \cdot \frac{1}{2}(-\lambda_{opt}) \cdot l_x = -\frac{u_x \cdot l_x}{2(u_x - l_x)}$.

Поскольку параллелограмм — это частный случай зонотопа, то выходное значение функции ReLU можно представить в виде зонотопа

$$\hat{y} = \begin{cases} \hat{x}, \text{ если } l_x > 0 \\ [0, 0], \text{ если } u_x \le 0 \\ [\lambda_l, \lambda_u] \cdot \hat{x} + [\mu_l, \mu_u] + [\mu_l, \mu_u] \cdot \epsilon_{\text{new}} \text{ , иначе} \end{cases}$$

Здесь $\epsilon_{new} \subseteq [-1; 1]$. Под λ_l и λ_u подразумевается приближение λ_{opt} с точностью до второго знака числом с плавающей запятой. Аналогично μ_l и μ_u — приближение μ_{opt}.

Самое большое преимущество зонотопов — приближение сигмоиды и гиперболического тангенса, проводится аналогочным образом через нахождение минимума площади построенного параллелограмма и построение зонотопа через «сдвиг» нижней границы к центру:

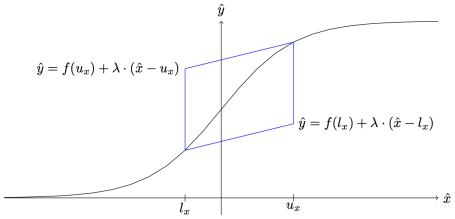


Рис. 14. Приближение сигмоиды с помощью зонотопа [14]

$$\hat{y} = \begin{cases} [f(u_x)_l, f(u_x)_u], \text{ если } l_x = u_x \\ [\lambda_l, \lambda_u] \cdot \hat{x} + [\mu_l^1, \mu_u^1] + [\mu_l^2, \mu_u^2] \cdot \epsilon_{\text{new}}, \quad \text{иначе} \end{cases}$$

Здесь $\lambda_{opt} = min(f'(l_x)), f'(u_x)), \mu_1 = \frac{1}{2}(f(u_x) + f(l_x) - \lambda_{opt}(u_x + l_x)), \mu_2 = \frac{1}{2} f(u_x) - f(l_x) - \lambda_{opt}(u_x - l_x))$

Поскольку теперь на следующий нейрон подаётся зонотоп, то после применения функции ReLU встаёт задача разбиения входного зонотопа на два, первый из которых отвечает за отрицательные значения, а второй — за положительные. Зонотоп, отвечающий за отрицательные входные значения, функция ReLU занулит, то есть спроецирует на вертикальную ось. Зонотоп, отвечающий за положительные значения, останется неизменным.

Следующим логичным шагом ставится задача объединения полученного отрезка и полученного зонотопа. Самый простой способ заключается в простом объединении, на рисунке 15 рассмотрен крайне удачный пример, когда результат объединения отрезка z_5 и зонотопа z_4 снова становится зонотопом z_6.

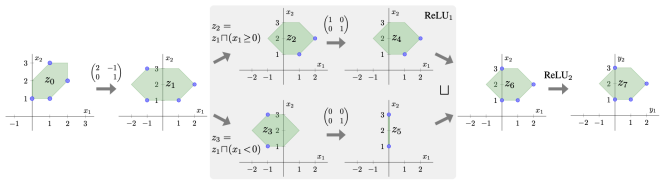


Рис. 15. Простое объединение зонотопов на примере алгоритма AI^2 [12]

На следующем примере [16] уже не получается получить зонотоп при помощи простого объединения, рассматривается идея объединения данных с сохранением формы зонотопа рассматриваемой области. То есть используется алгоритм объединения двух зонотопов в один, описанный выше на рисунке 12

Нейронная сеть задаётся следующим образом: $\mathcal{N}(x) =$

$$\begin{bmatrix} 1 & 1.1 \\ -1 & 1 \end{bmatrix} \text{ReLU}\left(\begin{bmatrix} 1 & -3 \\ 0 & 3 \end{bmatrix} x + \begin{bmatrix} 1 \\ 1 \end{bmatrix}\right) + \begin{bmatrix} -3 \\ 1.2 \end{bmatrix}$$

Проверяется свойство, что все $x \in [0;1]^2$ она будет относить ко второму классу.

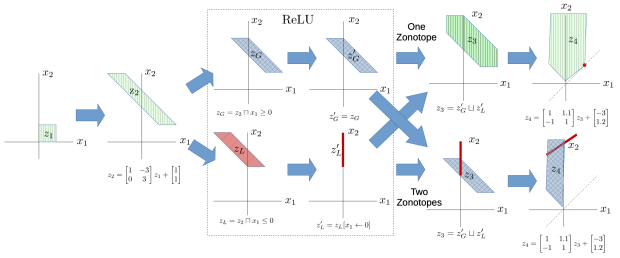


Рис. 16. Пример применения принципа объединения зонотопов [16]

После применения функции ReLU получаются два множества — зонотоп z'_G (сверху) и отрезок z'_L (снизу). Объединяя их при помощи обычного метода, получается просто фигура вида параллелограмм с «торчащим» из него отрезком, которая на следующем шаге сохраняет свою невыпуклость и не может отследить контрпример — точку (1,2;1,2), в которой нарушается свойство робастности. То есть нейронная сеть в данной точке принимает решение отнести входные данные к первому классу. Метод объединения зонотопов, результатом которого снова является зонотоп, точку (1,2;1,2) отслеживает.

E. Абстракция с применением star sets, звёздных множеств

Продолжением идеи зонотопа, заключающейся в том, чтобы сохранить линейное представление заданного множества, является star set или звёздное множество. Метод является основой алгоритма nnennum [17]

В статье [15] приводится следующее определение:

Обобщённым звёздным множеством называется тройка $\Theta = \langle x_0, V, P\rangle$, где $x_0 \in \mathbb{R}^n$ — центр этого множества, $V = \{v_1, v_2, \dots v_m\}$ — множество из $m(\leq n)$ векторов $\mathbb{R}^n$, называемое базисом, и $P : \mathbb{R}^n \to \{\top, \bot\}$ - предикат. Обобщённое звёздное множество Θ является подмножеством $\mathbb{R}^n$:

$$\Theta = \left\{x \mid \exists \bar{\alpha} = [\alpha_1, \dots, \alpha_m]^T,\right.$$

$$\left.\text{такое что } x = x_0 + \sum_{i=1}^{n} \alpha_i v_i \text{ и } P(\bar{\alpha}) = \top\right\}.$$

Другими словами, существуют такие константы α_i, «растягивающие» базисные векторы v_i по каждому направлению, с заданными ограничениями на эти константы $P(\bar{\alpha}) = \top$, что задаваемое звёздное множество будет центрально-симметричным и при применении аффинных преобразований будет сохранять свой вид. Если такое множество подать на вход нейронной сети, то выходное множество снова будет записано в таком же виде, что очень удобно для алгоритмов формальной верификации, использующих LP-программирование.

Стоит отметить, что зонотоп — частный случай звёздного множества, просто ограничения на константы будут отрезком $[0;1]$ или $[-1;1]$, что также можно задать в виде неравенства с соответствующей матрицей $C\alpha_i \leq 1$. Рассмотрим эту идею на примере.

Пример задания 2-мерного плоского множества (в $\mathbb{R}^2$) как зонотопа и как звёздного множества:

Базисные векторы $V = \{[1,0]^T, [0,1]^T\}$ и центр $x_0 = (3,3)$. Далее вводятся ограничения $g = [1,1,1,1]^T$ и $P(\bar{\alpha}) = C\bar{\alpha} \leq g$, где $C = \begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 \end{bmatrix}^T$ (несложно убедиться, что такая запись ограничений аналогична системе неравенств $\alpha_1 \leq 1, -\alpha_1 \leq 1, \alpha_2 \leq 1, -\alpha_2 \leq 1$ или $\alpha_1, \alpha_2 \in [-1;1]$).

Тогда звёздное множество $\Theta = \langle x_0, V, P\rangle$ определяет прямоугольник, который можно записать в виде конъюнкции линейных неравенств

$$\Theta = \{(x,y) \mid 2 \leq x \leq 4 \wedge 2 \leq y \leq 4\}$$

или в виде зонотопа $\{(3+\epsilon_1; 3+\epsilon_2) \mid \epsilon_1, \epsilon_2 \in [-1;1]\}$.

С другой стороны, мы можем задать ограничения на константы как $P(\bar{\alpha}) = (\alpha_1 - 3)^2 + (\alpha_2 - 3)^2 \leq 1$ и получим круг единичного радиуса с центром в точке $(3,3)$.

После применения афинных преобразований обе фигуры (прямоугольник и круг) переходят в фигуры того же вида с одним и тем же центром и одними и теми же и базисными векторами. Отдельно отметим, что левая фигура является аналогом сравнения по норме L_∞, а правая — по норме L_2. И замена норм не влияет на изменение центра и базисных векторов.

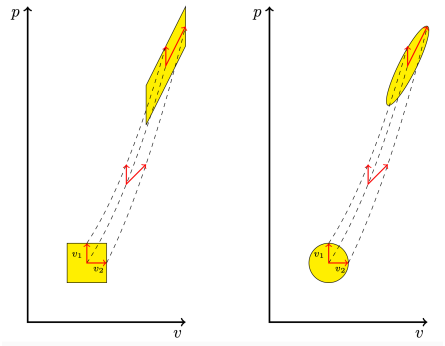


Рис. 17. Задание и изменение звёздных множеств [15]

Звёздное множество является минимальной по площади фигурой, приближающей функцию ReLU.

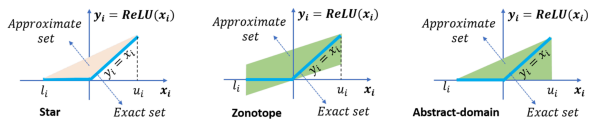


Рис. 18. Приближение функции ReLU звёздным множеством, зонотопом и абстрактным множеством [15]

IV. Заключение

В статье приведены основные математические методы, позволяющие существующим алгоритмам, основанным на идее достижимости каждого нейрона в нейронной сети, осуществлять верификацию работы нейронной сети. Дальнейшая работа подразумевает более глубокое исследование алгоритмов и возможно, создание новых, более эффективных алгоритмов.

V. Благодарности

Мы благодарны сотрудникам кафедры Информационной безопасности факультета ВМК МГУ имени М.В. Ломоносова за ценные обсуждения данной работы. Исследование выполнено при поддержке Междисциплинарной научно-образовательной школы Московского университета «Мозг, когнитивные системы, искусственный интеллект» Статья является продолжением серии публикаций, посвященных устойчивым моделям машинного обучения и кибербезопасности систем искусственного интеллекта [18], [19], [20]. Она подготовлена в рамках проекта кафедры Информационной безопасности факультета ВМК МГУ имени М.В. Ломоносова по созданию и развитию магистерской программы ”Искусственный интеллект в кибербезопасности”[21].

Список литературы

[1] Muhammad Shafique Mahum Naseer Theocharis Theocharides. Robust Machine Learning Systems: Challenges,Current Trends, Perspectives, and the Road Ahead. — 2020. — . — URL: https://ieeexplore.ieee.org/document/8979377.

[2] Safety verification of deep neural networks / Xiaowei Huang, Marta Kwiatkowska, Sen Wang, Min Wu. — 2016. — URL: https://arxiv.org/abs/1610.06940.

[3] Ehlers Ruediger. Formal verification of piece-wise linear feed-forward neural networks. — 2017. — URL: https://arxiv.org/abs/1705.01320.

[4] Shriver David, Elbaum Sebastian, Dwyer Matthew B. Dnnv: A framework for deep neural network verification. — 2021. — URL: https://arxiv.org/abs/2105.12841.

[5] Verification of deep convolutional neural networks using imagestars / Hoang-Dung Tran, Stanley Bak, Weiming Xiang, Taylor T. Johnson. — 2020. — URL: https://arxiv.org/abs/2004.05511.

[6] Tjeng Vincent, Xiao Kai, Tedrake Russ. Evaluating robustness of neural networks with mixed integer programming. — 2017. — URL: https://arxiv.org/abs/1711.07356.

[7] A dual approach to scalable verification of deep networks / Krishnamurthy, Dvijotham, Robert Stanforth et al. — 2018. — URL: https://arxiv.org/abs/1803.06567.

[8] Boosting robustness certification of neural networks / Gagandeep Singh, Timon Gehr, Markus Püschel, Martin Vechev // International Conference on Learning Representations. — 2019. — URL: https://openreview.net/forum?id=HJgeEh09KQ.

[9] Xiang Weiming, Tran Hoang-Dung, Johnson Taylor T. Specification-guided safety verification for feedforward neural networks. — 2018. — URL: https://arxiv.org/abs/1812.06161.

[10] Formal security analysis of neural networks using symbolic intervals / Shiqi Wang, Kexin Pei, Justin Whitehouse et al. — 2018. — URL: https://arxiv.org/abs/1804.10829.

[11] Efficient formal safety analysis of neural networks / Shiqi Wang, Kexin Pei, Justin Whitehouse et al. — 2018. — URL: https://arxiv.org/abs/1809.08098.

[12] Timon Gehr Matthew Mirman Dana Drachsler-Cohen. AI2: Safety and Robustness Certification of Neural networks with Abstract Interpretation. — 2018. — URL: https://files.sri.inf.ethz.ch/website/papers/sp2018.pdf.

[13] Ghorbal Khalil, Goubault Eric, Putot Sylvie. The zonotope abstract domain taylor1+. — 2009. — URL: https://link.springer.com/chapter/10.1007/978-3-642-02658-4_47#chapter-info.

[14] Fast and effective robustness certification / Gagandeep Singh, Timon Gehr, Matthew Mirman et al. // Advances in Neural Information Processing Systems / Ed. by S. Bengio, H. Wallach, H. Larochelle et al. — Vol. 31. — Curran Associates, Inc., 2018. — URL: https://proceedings.neurips.cc/paper/2018/file/f2f446980d8e971ef3da97af089481c3-Paper.pdf.

[15] Duggirala Parasara Sridhar, Viswanathan Mahesh. Parsimonious, simulation based verification of linear systems. — 2016. — URL: https://www.cs.unc.edu/~psd/files/research/2016/psd-mv-cav-16.pdf.

[16] Optimization and abstraction: a synergistic approach for analyzing neural network robustness / Greg Anderson, Shankara Pailoor, Isil Dillig, Swarat Chaudhuri. — ACM, 2019. — jun. — URL: https://doi.org/10.1145/3314221.3314614.

[17] Improved geometric path enumeration for verifying relu neural networks / Stanley Bak, Hoang-Dung Tran, Kerianne Hobbs, Taylor T. Johnson. — 2020. — URL: https://link.springer.com/chapter/10.1007/978-3-030-53288-8_4.

[18] Namiot Dmitry Eugene Ilyushin, Pilipenko Oleg. On trusted ai platforms. — 2022. — URL: http://injoit.org/index.php/j1/article/view/1398.

[19] Namiot Dmitry Eugene Ilyushin, Chizhov Ivan. Attacks on machine learning systems-common problems and methods. — 2022. — URL: http://injoit.org/index.php/j1/article/view/1398.

[20] Dmitry Namiot, Ilyushin Eugene. Generative models in machine learning. — 2022. — URL: http://injoit.org/index.php/j1/article/view/1398.

[21] Iskusstvennyj intellekt v kiberbezopasnosti. — Provereno: 08.08.2022. — URL: https://cs.msu.ru/node/3732.

Е.Н. Строева - МГУ имени М.В. Ломоносова (email: katestroeva@gmail.com)

А.А. Тонких - МГУ имени М.В. Ломоносова (email: alexej-t@mail.ru)

Methods for Formal Verification of Artificial Neural Networks: A Review of Existing Approaches

Ekaterina Stroeva, Aleksey Tonkikh

***Abstract*—Among recent publications containing an overview and systematization of algorithms for formal verification of neural networks, a classification of algorithms is proposed which is based on three following properties: reachability, optimization, and search. Reachability-based methods select data from a range of input values based on predefined constraints on the input values, and approximate this set using symbolic mathematical constructs. The main problem of this approach is overapproximation, i.e., the output set is too wide. Another big problem is preservation of linearity after applying the ReLU activation function, which is necessary for applying the backpropagation algorithm. In present article we analyze, compare, and sistemathize mathematical constructions and attempts to approximate the point data by flat continuous sets with the least possible approximation. We also discuss the most effective methods of solving the problem of applying the ReLU activation function to symbolic elements.**

***Keywords*—formal verification algorithms**

Список литературы

[1] Muhammad Shafique Mahum Naseer Theocharis Theocharides. Robust Machine Learning Systems: Challenges,Current Trends, Perspectives, and the Road Ahead. — 2020. — . — URL: https://ieeexplore.ieee.org/document/8979377.

[2] Safety verification of deep neural networks / Xiaowei Huang, Marta Kwiatkowska, Sen Wang, Min Wu. — 2016. — URL: https://arxiv.org/abs/1610.06940.

[3] Ehlers Ruediger. Formal verification of piece-wise linear feed-forward neural networks. — 2017. — URL: https://arxiv.org/abs/1705.01320.

[4] Shriver David, Elbaum Sebastian, Dwyer Matthew B. Dnnv: A framework for deep neural network verification. — 2021. — URL: https://arxiv.org/abs/2105.12841.

[5] Verification of deep convolutional neural networks using imagestars / Hoang-Dung Tran, Stanley Bak, Weiming Xiang, Taylor T. Johnson. — 2020. — URL: https://arxiv.org/abs/2004.05511.

[6] Tjeng Vincent, Xiao Kai, Tedrake Russ. Evaluating robustness of neural networks with mixed integer programming. — 2017. — URL: https://arxiv.org/abs/1711.07356.

[7] A dual approach to scalable verification of deep networks / Krishnamurthy, Dvijotham, Robert Stanforth et al. — 2018. — URL: https://arxiv.org/abs/1803.06567.

[8] Boosting robustness certification of neural networks / Gagandeep Singh, Timon Gehr, Markus Püschel, Martin Vechev // International Conference on Learning Representations. — 2019. — URL: https://openreview.net/forum?id=HJgeEh09KQ.

[9] Xiang Weiming, Tran Hoang-Dung, Johnson Taylor T. Specification-guided safety verification for feedforward neural networks. — 2018. — URL: https://arxiv.org/abs/1812.06161.

[10] Formal security analysis of neural networks using symbolic intervals / Shiqi Wang, Kexin Pei, Justin Whitehouse et al. — 2018. — URL: https://arxiv.org/abs/1804.10829.

[11] Efficient formal safety analysis of neural networks / Shiqi Wang, Kexin Pei, Justin Whitehouse et al. — 2018. — URL: https://arxiv.org/abs/1809.08098.

[12] Timon Gehr Matthew Mirman Dana Drachsler-Cohen. AI2: Safety and Robustness Certification of Neural networks with Abstract Interpretation. — 2018. — URL: https://files.sri.inf.ethz.ch/website/papers/sp2018.pdf.

[13] Ghorbal Khalil, Goubault Eric, Putot Sylvie. The zonotope abstract domain taylor1+. — 2009. — URL: https://link.springer.com/chapter/10.1007/978-3-642-02658-4_47#chapter-info.

[14] Fast and effective robustness certification / Gagandeep Singh, Timon Gehr, Matthew Mirman et al. // Advances in Neural Information Processing Systems / Ed. by S. Bengio, H. Wallach, H. Larochelle et al. — Vol. 31. — Curran Associates, Inc., 2018. — URL: https://proceedings.neurips.cc/paper/2018/file/f2f446980d8e971ef3da97af089481c3-Paper.pdf.

[15] Duggirala Parasara Sridhar, Viswanathan Mahesh. Parsimonious, simulation based verification of linear systems. — 2016. — URL: https://www.cs.unc.edu/~psd/files/research/2016/psd-mv-cav-16.pdf.

[16] Optimization and abstraction: a synergistic approach for analyzing neural network robustness / Greg Anderson, Shankara Pailoor, Isil Dillig, Swarat Chaudhuri. — ACM, 2019. — jun. — URL: https://doi.org/10.1145/3314221.3314614.

[17] Improved geometric path enumeration for verifying relu neural networks / Stanley Bak, Hoang-Dung Tran, Kerianne Hobbs, Taylor T. Johnson. — 2020. — URL: https://link.springer.com/chapter/10.1007/978-3-030-53288-8_4.

[18] Namiot Dmitry Eugene Ilyushin, Pilipenko Oleg. On trusted ai platforms. — 2022. — URL: http://injoit.org/index.php/j1/article/view/1398.

[19] Namiot Dmitry Eugene Ilyushin, Chizhov Ivan. Attacks on machine learning systems-common problems and methods. — 2022. — URL: http://injoit.org/index.php/j1/article/view/1398.

[20] Dmitry Namiot, Ilyushin Eugene. Generative models in machine learning. — 2022. — URL: http://injoit.org/index.php/j1/article/view/1398.

[21] Iskusstvennyj intellekt v kiberbezopasnosti. — Provereno: 08.08.2022. — URL: https://cs.msu.ru/node/3732.