УДК 004.654: 004.852

Т.М. Буланая, Е.В. Воронюк

РАЗРАБОТКА НЕЙРОИНФОРМАЦИОННОЙ ТЕХНОЛОГИИ ДЛЯ АНАЛИЗА ДАННЫХ МОНИТОРИРОВАНИЯ

Аннотация. В рамках этой работы: была разработана и реализована библиотека для работы с нейронными сетями - Java Neural Network Modeling Framework. Был исследован и реализован алгоритм классификации - Каскадная корреляция, был разработан и реализован программный комплекс с графическим интерфейсом, который позволяет обучать нейронные сети, наблюдать и оценивать процесс обучения, хранить сети на жестком диске для дальнейшего использования.

Ключевіе слова: Нейронная сеть, каскадная корреляция, JAVA, JNMF.

Для решения задач анализа накопленных данных можно выделить два класса технологий. На первых стадиях информатизации всегда требуется навести порядок именно в процессах повседневной рутинной обработки данных (накопление, информационный поиск, установления факта наличия/отсутствия зависимости между данными), на что, и ориентированы технологии первичной обработки данных (например, MS Excel [1]). Технологии второго класса — технологии интеллектуального анализа данных Data Mining (например, нейроимитатор NeuroShell 2 [2]), являются вторичными и дорогостоящими, по отношению к технология с обработки данных.

Цель работы – разработать гибкую нейроинформационную технологию, которая позволит обобщать и анализировать знания, накопленные экспериментальным путем.

Этапы анализа данных в разработанной нейроинформационной технологии MiningLibs включают: кодирование входов-выходов нейросети (нейросеть работает только с числами); нормирование данных (результаты нейроанализа не должны зависеть от выбора единиц из-

мерения); обучение нескольких нейронных сетей с различной архитектурой (результат обучения зависит, как от размеров сети, так и от её начальной конфигурации); отбор оптимальных сетей, тех которые дадут наименьшую ошибку предсказания на неизвестных данных; оценка значимости и ошибки предсказаний. На рисунке 1 представлена структурная схема нейроинформационной технологии MiningLibs.

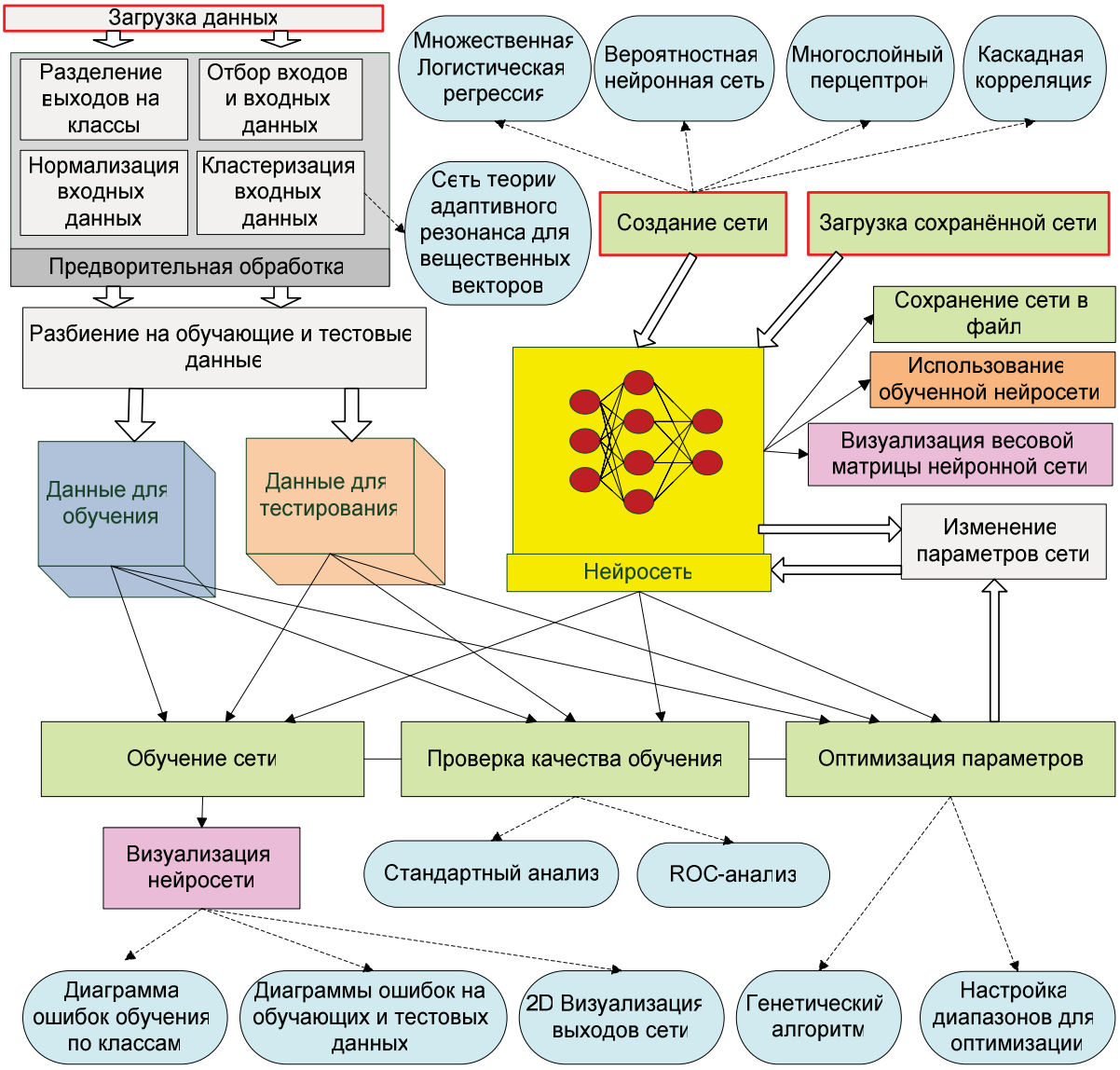

Рисунок 1 - Структурная схема нейроинформационной технологии для анализа данных мониторирования

Программной реализацией на языке java, предложенной нейроинформационной технологии MiningLibs, является библиотека Java Neural Network Modeling Framework (JNMF)

 ISSN 1562-9945

(http://sourceforge.net/projects/jnmf-new-gui/). Разработанная библиотека JNMF удобная в использовании, она позволяет конструировать нейронные сети различной сложности.

В JNMF для работы с нейросетями предусмотренные следующие операции:

- Конфигурирование:
- Изменение узловых элементов сети (нейронов);
- Изменение связей;
- Изменение алгоритмов сети.
- Операции жизненного цикла обучения / работы:
- Ввод данных;
- Начало обучения / работы;
- Наблюдение за процессом обучения / работы;
- Получение результатов / повтор обучения.

Отличительной чертой нейроинформационной технологии MiningLibs библиотеки JNMF, является наличие реализации алгоритма каскадной корреляции [3].

Алгоритм. Построение архитектуры каскадной корреляции

На вход алгоритма подаются:

1) Параметры:

- E_{min} - минимальная допустимая погрешность сети. Сеть учится до тех пор, пока реальная погрешность не станет меньше этого значения.

- minOutputsLearningSpeed - минимальная скорость обучения исходным нейронам и нейронам кандидатов. Если реальная скорость обучения падает ниже этого значения, то нужно завершить обучение.

- minCorrelationFluctuation - минимальная скорость изменения корреляции нейронов кандидатов. Если реальная скорость обучения падает ниже этого значения, то нужно завершить обучение кандидатам и выбрать победителя.

- candidatesCount – число нейронов кандидатов которые соревнуются.

2) Обучающие данные:

Данные состоят из набора обучающих пар. Каждая i-ая обучающая пара это пара векторов $\{x_i, y_i\}$, где $x_i = \{x_{ij} | j = 1,2,..,n\}$ - входной вектор; $y_i = \{y_{ij} | j = 1,2,..,m\}$ – ожидаемый исходный вектор; Пусть k - количество обучающих пар, тогда обозначим все входные данные матрицей следующего вида:

$$L = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} & y_{11} & y_{12} & \cdots & y_{1m} \\ x_{21} & x_{22} & \cdots & x_{2n} & y_{21} & y_{22} & \cdots & y_{2m} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{k1} & x_{k2} & \cdots & x_{kn} & y_{k1} & y_{k2} & \cdots & y_{km} \end{pmatrix}.$$

Обучающие пары расположены в сроках этой матрицы.

Тогда $L_{ij} = \begin{cases} x_{ij}, & \text{якщо } 1 \le j \le n \\ y_{ij-n}, & \text{якщо } n+1 \le j \le n+m \end{cases}$

Нормирование данных

1. Нормировать обучающие данные

1.1. Вычислить минимальные и максимальные значения в каждом столбцы матрицы L: $min_j = min_{i=1..k}(L_{ij})$; $max_j = max_{i=1..k}(L_{ij})$, $j = 1..n+m$

1.2. Привести все элементы матрицы L к интервалу [-1, 1]. $L^н_{ij} = -1 + \frac{2(L_{ij} - min_j)}{max_j - min_j}$, $i = 1..k$; $j = 1..n+m$. В результате получим нормированные данные $L^н$.

Построение начальной структуры сети (рис. 2)

2. Построить начальный вид сети (2 слоя), которая может работать с обучающими данными.

2.1. Построить первый слоя сети. Число нейронов в нем будет равнять n - размера входного вектора данных. Активационная функция вида $f(x) = x$

2.2. Создать второй (он же последний) слой сети. Число нейронов в нем будет равнять m – размера исходного вектора данных. Активационная функция, суживающая в диапазоне [-1, 1]: $f(x) = \tanh(x)$

 ISSN 1562-9945

2.3. Попарно связать все нейроны из первого слоя с нейронами последнего слоя.

Обучение исходным нейронам сети (эпоха t)

3. Непосредственное обучение синапсов исходных нейронов. Для каждой обучающей пары $L^{н}_{p} = \{x_p, y_p\}$ данных $L^{н(t)}$ (где t - номер текущей эпохи обучения входным нейронам сети) осуществить шаг обработки сети:

3.1. Подать входной вектор x_i следующей учебной пары $L^{н}_{p}$ на вход сети.

3.2. Обработать сетью входной вектор.

3.3. Для каждого нейрона в исходном слое выполнить:

3.3.1. Собрать и сохранить информацию о данном исходном нейроне:

3.3.1.1. Сохранить уровень активации нейрона на данном шаге o_{pj}.

3.3.1.2. Сохранить величину net нейрона на данном шаге.

3.3.1.3. Сохранить погрешность нейрона на данном шаге [$e_{pj} = (o_{pj} - y_{pj})f'(net)$ где p – номер обучающей пары (или номер шага), j – номер исходного нейрона, o_{pj} – активация j-го исходного нейрона на p-том шаге; $f'(net)$ – значение производной от активационной функции нейрона в точке net].

3.3.1.4. Сохранить величину сигнала всех синапсов на данном шаге, которые входят в данный нейрон.

3.3.2. Научить все синапсы, что входят в данный исходный нейрон, методом Back Propagation или Quick Propagation [4] .

3.4. Если еще остались не обработанные обучающие пары (т.е. $p < k$), тогда перейти к шагу 3.1.

4. Вычисление погрешности сети. Вычислить общую погрешность сети за формулой: $E^{(t)} = \frac{1}{2k}\sum_p \sum_j (o_{pj} - y_{pj})^2$, где p – номер обу-

чающей пары (или номер шага). j – номер исходного нейрона. o_{pj} – активация j-го исходного нейрона на p-том шаге.

5. Проверка условия о достаточной минимизации погрешности. Если $E^{(t)} < E_{min}$ это перейти к шагу 16.

6. Получение новой последовательности обучающих пар. Перемешать все обучающие пары случайным чином. Иначе говоря, поменять местами строки матрицы $L^{H(t)}$ произвольным образом, не изменяя положение элементов в строке и не изменяя собственных значений элементов. Запишем такое преобразование как $L^{H(t+1)} = \text{random\_rows\_positions}\left(L^{H(t)}\right)$. Делаем это для того, чтобы полученная сеть не была привязана до одной конкретной последовательности обучающих пар.

7. Проверка скорости обучения. Если $\left|E^{(t)} - E^{(t-1)}\right| > \text{minLearningSpeed}$, где t – номер эпохи; $E^{(t)}$ - текущая общая погрешность сети; $E^{(t-1)}$ – предыдущая общая погрешность сети (на предыдущей эпохе), ИЛИ если $E^{(t-1)}$ не определено (текущая эпоха, есть первой), то перейти к шагу 4.

Подготовка к этапу обучения нейронам кандидатов (рис 3.)

8. Создание нового слоя. Добавить новый слой, перед исходным слоем.

9. Добавление нейронов кандидатов. Добавить в новый слой candidatesCount нейронов кандидатов. Активационная функция $f(x) = \tanh(x)$.

10. Соединение нейронов кандидатов с другими нейронами сети. Для каждого нейрона кандидата соединить его синапсами со всеми нейронами из предыдущих слоёв (с нейронами как первого(входного) слоя, так и с нейронами, которые были добавлены в процессе строительства сети). Вес каждого синапсу берется случайно в диапазоне [-1, 1].

 ISSN 1562-9945

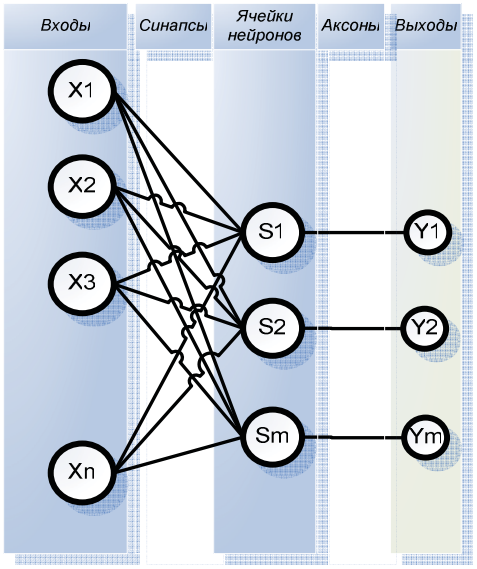

Рисунок 2 - Начальная структура НС каскадная корреляция

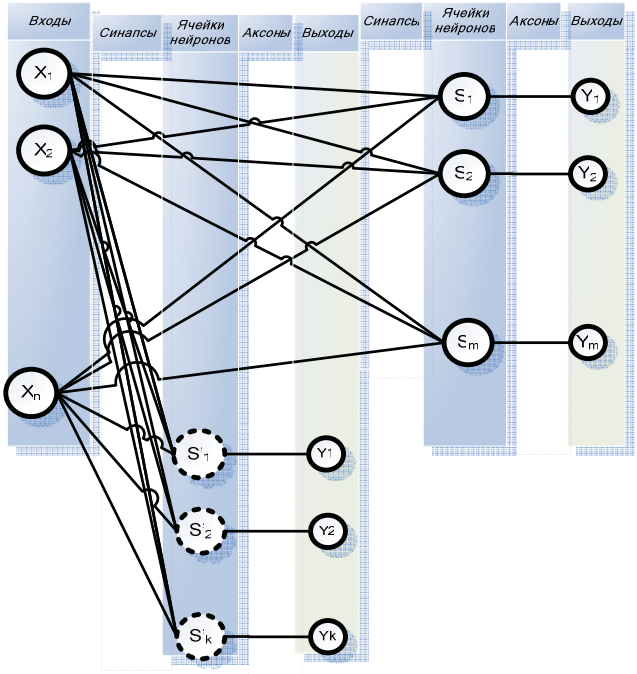

Рисунок 3 - Соревнования нейронов кандидатов

Организуется новое соревнование. Опять будет отобран и додан в сеть нейрон, чья корреляция была максимальной

Обучение нейронам кандидатов (эпоха обучения нейронам кандидатов). Прогонка обучающих пар и сбор информации о нейронах кандидатов и исходных нейронов. Для каждой обучающей пары $L^{н}_{p} = \{x$ данных (где t - номер текущей эпохи обучения входным нейронам сети) выполнить шаг обработки сети:

10.1. Подать входной вектор x_l следующей обучающей пары $L^{н}_{p}$ на вход сети.

10.2. Обработать сетью входной вектор.

10.3. Собрать и сохранить информацию о каждого нейрона кандидата:

10.3.1. Сохранить уровень активации нейрона на данном шаге.

10.3.2. Сохранить величину net нейрона на данном шаге.

10.3.3. Сохранить величину сигнала всех синапсов на данном шаге, которые входят в данный нейрон.

10.4. Для каждого нейрона в исходном слое выполнить:

10.4.1. Собрать и сохранить информацию о данном исходном нейроне:

10.4.1.1. Сохранить уровень активации нейрона на данном шаге.

10.4.1.2. Сохранить величину net нейрона на данном шаге.

10.4.1.3. Сохранить погрешность нейрона на данном шаге [$e_{pj} = (o_{pj} - y_{pj})f'(net)$ где p – номер шага, j – номер исходного нейрона, o_{pj} – активация j-го исходного нейрона на p-том шаге; $f'(net)$ – значение производной вот активационной функции нейрона в точке net].

10.4.1.4. Сохранить величину сигнала всех синапсов на данном шаге, которые входят в данный нейрон.

10.5. Если еще остались необработанные обучающие пары (т.е. $p < k$), тогда перейти к шагу 11.1.

11. Вычисление корреляции для каждого кандидата. Для каждого нейрона кандидата вычислить величину корреляции за формулой: $C_k^{(t^*)} = \sum_j \left| \sum_p o_{pk}(e_{pj} - \bar{e}_j) \right|$, где t^* – номер эпохи; k – номер нейрона кандидата; p – номер шага; j – номер исходного нейрона; o_{pk} – активация k-го нейрона кандидата на p-том шаге; $e_{pj} = (o_{pj} - y_{pj})f'(net)$ - погрешность j-го нейрона на p-том шаге (параметры этой формулы уже были описаны выше); $\bar{e}_j = \frac{1}{p}\sum_p e_{pj}$ - средняя погрешность j-го нейрона по всему шагам.

12. Вычисление максимальной корреляции. $C_{max}^{(t^*)} = \max_k C_k^{(t^*)}$

13. Проверка скорости роста корреляции. Если $\left| C_{max}^{(t^*)} - C_{max}^{(t^*-1)} \right| > \text{minCorrelationFluctuation}$, $C_{max}^{(t^*)}$ - текущее значение максимальной корреляции (на данной эпохе) $C_{max}^{(t^*-1)}$ – предыдущее значение максимальной корреляции (на предыдущей эпохе),

ИЛИ если $C_{max}^{(t'-1)}$ не определено (текущая эпоха обучения кандидатам, есть первой), то перейти к шагу 14.

Завершение этапа обучения нейронам кандидатов (выбор победителя) (рис.4.)

14. Перевести сеть в состояние «обучение синапсов исходных нейронов».

14.1. Удалить всех нейронов кандидатов кроме победителя (с максимальной корреляцией).

14.2. Соединить выход кандидата победителя с каждым из нейронов исходного слоя. Вес синапсов $\omega = 1$ (рис. 5.).

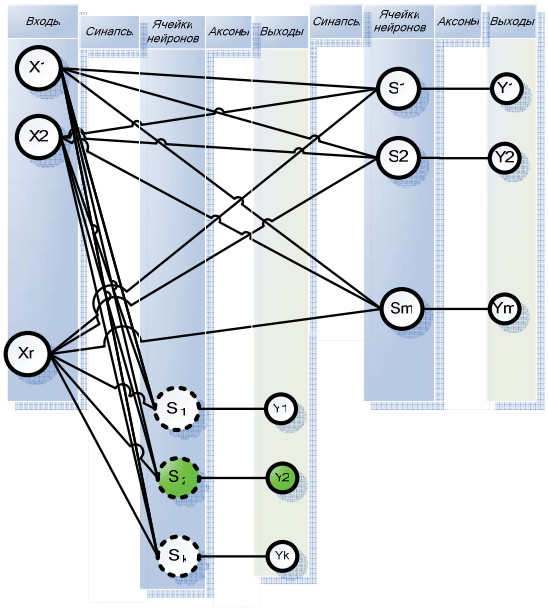

Рисунок 4 - Победителем выбран нейрон, который показал наибольшую корреляцию

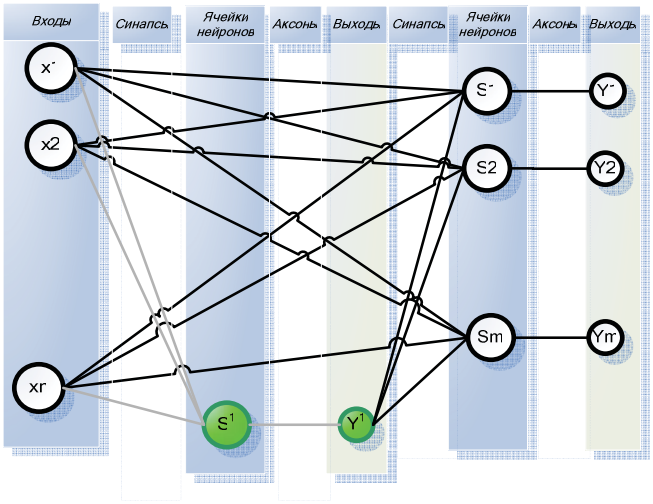

Рисунок 5 - Победитель связывается с выходным слоем, а его входные веса «замораживаются», а остальные кандидаты отбрасываются

14.3. Перейти к шагу 3.

15. Непосредственное обучение синапсов нейронов кандидатов. Для каждого нейрона кандидата выполнить:

15.1. Для каждого входного синапсу вычислить величину изменения веса: $\Delta\omega_i = \sum_p(\delta_p I_{pi})$, где i - номер синапсу; I_{pi} – сигнал i-го синапсу на p-том шаге; $\delta_p = \sum_j \mathrm{sign}(\sum_p o_{pk}(e_{pj} - \bar{e}_j))(e_{pj} - \bar{e}_j)f'(net)$, где k – номер нейрона кандидата; sign – функция, которая возвращает -1

если значение аргумента < 0 или 1 в противоположном случае; net – величина net нейрона кандидата; $f'(net)$ – значение производной активационной функции нейрона кандидата; e_{pj} $\bar{e}_j$ - уже были описаны выше.

15.2. Изменить вес синапсу $\omega_i^{(n+1)} = \omega_i^{(n)} + \Delta\omega_i^{(n)}$, где n - номер эпохи.

16. Перейти к шагу 9.

Конец

17. Сеть научена – КОНЕЦ

Следует отметить, что в алгоритме каскадной корреляции применялись методы обучения с учителем. Для оптимизации работы процедуры построения архитектуры каскадной корреляции, был реализован подбор параметров сети на базе генетических алгоритмов, а оценки качества модели каскадной корреляции применялся инструмент ROC-анализа.

Главные преимущества нейроинформационной технологии Miningibs библиотеки JNMF является:

- Мощность. Модель нейронной сети JNMF есть абстрактной. Она не привязана, к какой либо предметной области и не заточена под конкретный класс задач (пример, использования библиотеки[5,6]). Библиотека JNMF позволяет конструировать нейронные сети различной сложности, со статической или динамической структурой.
- Удобство. Если пользователь знаком с теорией нейронных сетей, то использовать библиотеку будет очень легко. Библиотека содержит набор стандартных компонент, предназначенных для решения типичных задач (например, алгоритм Обратного распространения).
- Гибкость. Библиотека JNMF построенная на принципах объектно-ориентированного программирования. Все ключевые компоненты (структурные компоненты, алгоритмы) абстрактные. Поэтому пользователь может писать свои компоненты и использовать их на месте базовых, тем самым расширяя функциональность библиотеки.

Теоретически, используя JNMF можно решить класс задач, которые подразумевают использование нейронных сетей.

ЛИТЕРАТУРА

1. www.office.microsoft.com
2. www.neuroshell.com
3. The Cascade-Correlation Learning Architecture Scott E. Fahlman and Christian Lebiere, CMU-CS-90-10, 1991.
4. Нейронные сети для обработки информации / С. Осовский. – М.: Финансы и статистика, 2004. – 344 с.
5. Застосування нейронної мережі в задачах аналізу газового оточення сенсорів / Т. М. Булана, Є. В. Воронюк, І. В. Гомілко, О. Ю. Ляшко, // Проблеми прикладної математики та ком'ютерних наук: тематична наук. конф. за підсумками наук.-досл. роботи ДНУ за 2007 – 2008рр.: тези доп. – Дніпропетровськ, 2008. – С. 22.
6. Анализ информативных акустических параметров при моделировании процесса струйного измельчения / Н.С. Прядко, Т.М. Буланая, Л.Ж. Горобец, В.Л. Баранов, Е.В. Воронюк, Р.А. Гавриленко // Системные технологии. – 2011. – Т.2. – с. 94 – 99.