6. Оценка и интерпретатор ответа

Эта глава посвящена обзору различных видов оценок, способам их вычисления. В ней так же рассмотрен способ определения уровня уверенности сети в выданном ответе и приведен способ построения оценок, позволяющих определять уровень уверенности. Приведен основной принцип проектирования оценки - надо учить сеть тому, что мы хотим от нее получить.

Напомним основные функции, которые должна выполнять оценка:

1. Вычислять оценку решения, выданного сетью.
2. Вычислять производные этой оценки по выходным сигналам сети.

Кроме оценок, в первом разделе этой главы рассмотрен другой, тесно связанный с ней объект - интерпретатор ответа. Основное назначение этого объекта - интерпретировать выходной вектор сети как ответ, понятный пользователю. Однако, при определенном построении интерпретатора и правильно построенной по нему оценке, интерпретатор ответа может также оценивать уровень уверенности сети в выданном ответе.

6.1 Интерпретатор ответа

Как было показано в главе «Описание нейронных сетей», ответ, выдаваемый нейронной сетью, как правило, является числом, из диапазона $[a,b]$. Если ответ выдается несколькими нейронами, то на выходе сети мы имеем вектор, каждый компонент которого лежит в интервале $[a,b]$. Если в качестве ответа требуется число из этого диапазона, то мы можем его получить. Однако, в большинстве случаев это не так. Достаточно часто требуемая в качестве ответа величина лежит в другом диапазоне. Например, при предсказании температуры воздуха 25 июня в Красноярске ответ должен лежать в интервале от 5 до 35 градусов Цельсия. Сеть не может дать на выходе такого сигнала. Значит, прежде чем обучать сеть необходимо решить в каком виде будем требовать ответ. В данном случае ответ можно требовать в виде $\alpha = (b-a)(T - T_{\min}) / (T_{\max} - T_{\min}) + a$, где T - требуемая температура, $T_{\min}$ и $T_{\max}$ минимальная и максимальная температуры, α - ответ, который будем требовать от сети. При интерпретации ответа необходимо проделать обратное преобразование. Если сеть выдала сигнал α, то ответом является величина $T = (\alpha - a)(T_{\max} - T_{\min}) / (b-a) + T_{\min}$. Таким образом, можно интерпретировать выдаваемый сетью сигнал, как величину из любого, наперед заданного диапазона.

Если при составлении обучающего множества ответ на примеры определялся с некоторой погрешностью, то от сети следует требовать не точного воспроизведения ответа, а попадания в интервал заданной ширины. В этом случае интерпретатор ответа может выдать сообщение о правильности (попадании в интервал) ответа.

Другим, часто встречающимся случаем, является предсказание сетью принадлежности входного вектора одному из заданных классов. Такие задачи называют задачами классификации, а решающие их сети - классификаторами. В простейшем случае задача классификации ставится следующим образом: пусть задано N классов. Тогда нейросеть выдает вектор из N сигналов. Однако, нет единого универсального правила интерпретации этого вектора. Наиболее часто используется интерпретация по максимуму: номер нейрона, выдавшего максимальный по величине сигнал, является номером класса, к которому относится предъявленный сети входной вектор. Такие интерпретаторы ответа называются интерпретаторами, кодирующими ответ **номером канала** (номер нейрона - номер класса). Все интерпретаторы, использующие кодирование номером канала, имеют один большой недостаток - для классификации на N классов требуется N выходных нейронов. При большом N требуется много выходных нейронов для получения ответа. Однако существуют и другие виды интерпретаторов.

Двоичный интерпретатор. Основная идея двоичного интерпретатора - получение на выходе нейронной сети двоичного кода номера класса. Это достигается двухэтапной интерпретацией:

1. Каждый выходной сигнал нейронной сети интерпретируется как 1, если он больше $(a+b)/2$, и как 0 в противном случае.
2. Полученная последовательность нулей и единиц интерпретируется как двоичное число.

Двоичный интерпретатор позволяет интерпретировать N выходных сигналов нейронной сети как номер одного из 2^N классов.

Порядковый интерпретатор. Порядковый интерпретатор кодирует номер класса подстановкой. Отсортируем вектор выходных сигналов по возрастанию. Вектор, составленный из номеров нейронов последовательно расположенных в отсортированном векторе выходных сигналов, будет подстанов-

кой. Если каждой подстановке приписать номер класса, то такой интерпретатор может закодировать *N!* классов используя *N* выходных сигналов.

6.2 Уровень уверенности

Часто при решении задач классификации с использованием нейронных сетей недостаточно простого ответа «входной вектор принадлежит *K-му* классу». Хотелось бы также оценить уровень уверенности в этом ответе. Для различных интерпретаторов вопрос определения уровня уверенности решается поразному. Однако, необходимо учесть, что от нейронной сети нельзя требовать больше того, чему ее обучили. В этом разделе будет рассмотрен вопрос об определении уровня уверенности для нескольких интерпретаторов, а в следующем будет показано, как построить оценку так, чтобы нейронная сеть позволяла его определить.

1. Кодирование номером канала. Знаковый интерпретатор. Знаковый интерпретатор работает в два этапа.

1. Каждый выходной сигнал нейронной сети интерпретируется как 1, если он больше $(a+b)/2$, и как 0 в противном случае.
2. Если в полученном векторе только одна единица, то номером класса считается номер нейрона, сигнал которого интерпретирован как 1. В противном случае ответом считается неопределенный номер класса (ответ «не знаю»).

Для того чтобы ввести уровень уверенности для этого интерпретатора потребуем, чтобы при обучении сети для всех примеров было верно неравенство: $\left|\alpha_i - (a+b)/2\right| \le \varepsilon$, где $i = 1,\text{K}\ ,N$; α_i - i-ый выходной сигнал. ε - уровень надежности (насколько сильно сигналы должны быть отделены от $(a+b)/2$ при обучении). В этом случае уровень уверенности R определяется следующим образом: $R = \min\left\{1; \min\dfrac{\left|\alpha_i - (a+b)/2\right|}{\varepsilon}\right\}$. Таким образом, при определенном ответе уровень уверенности показывает, насколько ответ далек от неопределенного, а в случае неопределенного ответа - насколько он далек от определенного.

2. Кодирование номером канала. Максимальный интерпретатор. Максимальный интерпретатор в качестве номера класса выдает номер нейрона, выдавшего максимальный сигнал. Для такого интерпретатора в качестве уровня уверенности естественно использовать некоторую функцию от разности между максимальным и вторым по величине сигналами. Для этого потребуем, чтобы при обучении для всех примеров обучающего множества разность между максимальным и вторым по величине сигналами была не меньше уровня надежности ε. В этом случае уровень уверенности вычисляется по следующей формуле: $R = \max\left\{1; (\alpha_i - \alpha_j)/\varepsilon\right\}$, где α_i - максимальный, а α_j - второй по величине сигналы.

3. Двоичный интерпретатор. Уровень надежности для двоичного интерпретатора вводится так же, как и для знакового интерпретатора при кодировании номером канала.

4. Порядковый интерпретатор. При использовании порядкового интерпретатора в качестве уровня уверенности естественно брать функцию от разности двух соседних сигналов в упорядоченном по возрастанию векторе выходных сигналов. Для этого потребуем, чтобы при обучении для всех примеров обучающего множества в упорядоченном по возрастанию векторе выходных сигналов разность между двумя соседними элементами была не меньше уровня надежности ε. В этом случае уровень уверенности можно вычислить по формуле $R = \min_i\left\{1; \left(\alpha_{i+1} - \alpha_i\right)/\varepsilon\right\}$, причем вектор выходных сигналов предполагается отсортированным по возрастанию.

В заключение заметим, что для ответа типа число, ввести уровень уверенности подобным образом невозможно. Пожалуй, единственным способом оценки достоверности результата является консилиум нескольких сетей - если несколько сетей обучены решению одной и той же задачи, то в качестве ответа можно выбрать среднее значение, а по отклонению ответов от среднего можно оценить достоверность результата.

6.3 Построение оценки по интерпретатору

Если в качестве ответа нейронная сеть должна выдать число, то естественной оценкой является квадрат разности выданного сетью выходного сигнала и правильного ответа. Все остальные оценки для обучения сетей решению таких задач являются модификациями данной. Приведем пример такой моди-

фикации. Пусть при составлении задачника величина $\overline{\alpha}$, являющаяся ответом, измерялась с некоторой точностью ε. Тогда нет смысла требовать от сети обучиться выдавать в качестве ответа именно величину $\overline{\alpha}$. Достаточно, если выданный сетью ответ попадет в интервал $\left[\overline{\alpha}-\varepsilon,\overline{\alpha}+\varepsilon\right]$. Оценка, удовлетворяющая этому требованию, имеет вид:

$$H=\begin{cases}0, & \text{при } \left|\alpha-\overline{\alpha}\right|\le\varepsilon,\\ \left(\alpha-\overline{\alpha}-\varepsilon\right)^2, & \text{при } \alpha>\overline{\alpha}+\varepsilon,\\ \left(\alpha-\overline{\alpha}+\varepsilon\right)^2, & \text{при } \alpha<\overline{\alpha}-\varepsilon.\end{cases}$$

Эту оценку будем называть оценкой числа с допуском ε.

Для задач классификации также можно пользоваться оценкой типа суммы квадратов отклонений выходных сигналов сети от требуемых ответов. Однако, эта оценка плоха тем, что во-первых, требования при обучении сети не совпадают с требованиями интерпретатора, во-вторых - такая оценка не позволяет оценить уровень уверенности сети в выданном ответе. Достоинством такой оценки является ее универсальность. Опыт работы с нейронными сетями, накопленный красноярской группой НейроКомп, свидетельствует о том, что при использовании оценки, построенной по интерпретатору, в несколько раз возрастает скорость обучения. Рассмотрим построение оценок по интерпретатору для четырех рассмотренных в предыдущем разделе интерпретаторов ответа.

1. Кодирование номером канала. Знаковый интерпретатор. Пусть для рассматриваемого примера правильным ответом является *k*-ый класс. Тогда вектор выходных сигналов сети должен удовлетворять следующей системе неравенств:

$$\begin{cases}\alpha_i<(a+b)/2-\varepsilon, & i\ne k\\ \alpha_k>(a+b)/2+\varepsilon,\end{cases}$$

где ε- уровень надежности.

Оценку, вычисляющую расстояние от точки α в пространстве выходных сигналов до множества точек, удовлетворяющих этой системе неравенств, можно записать в виде:

$$H=\sum_{i\ne k}\left(\alpha_i-(a+b)/2+\varepsilon\right)^2+\left(\alpha_k-(a+b)/2-\varepsilon\right)^2.$$

Производная оценки по i-му выходному сигналу равна $\dfrac{\partial H}{\partial\alpha_i}=\begin{cases}2(\alpha_i-(a+b)/2+\varepsilon), i\ne k\\ 2(\alpha_k-(a+b)/2-\varepsilon).\end{cases}$.

2. Кодирование номером канала. Максимальный интерпретатор. Пусть для рассматриваемого примера правильным ответом является *k*-ый класс. Тогда вектор выходных сигналов сети должен удовлетворять следующей системе неравенств: $\alpha_k-\varepsilon\ge\alpha_i$, при $i\ne k$. Оценкой решения сетью данного примера является расстояние от точки α в пространстве выходных сигналов до множества точек, удовлетворяющих этой системе неравенств. Для записи оценки, исключим из вектора выходных сигналов сигнал α_k, а остальные сигналы отсортируем по убыванию. Обозначим величину $\alpha_k-\varepsilon$ через β_0, а вектор отсортированных сигналов через $\beta_1\ge\beta_2\ge\text{K}\ge\beta_{N-1}$. Система неравенств в этом случае приобретает вид $\beta_0\ge\beta_i$, при $i>1$. Множество точек удовлетворяющих этой системе неравенств обозначим через $\boldsymbol{D}$. Очевидно, что если $\beta_0\ge\beta_1$, то точка β принадлежит множеству $\boldsymbol{D}$. Если $\beta_0<\beta_1$, то найдем проекцию точки β на гиперплоскость $\beta_0=\beta_1$. Эта точка имеет координаты $\beta^1=\left(\dfrac{\beta_0+\beta_1}{2},\dfrac{\beta_0+\beta_1}{2},\beta_2,\text{K},\beta_{N-1}\right)$. Если $\dfrac{\beta_0+\beta_1}{2}>\beta_2$, то точка β^1 принадлежит множеству $\boldsymbol{D}$. Если нет, то точку β нужно проектировать на гиперплоскость $\beta_0=\beta_1=\beta_2$. Найдем эту точку. Ее координаты можно записать в следующем виде $(b,b,b,\beta_3,\text{K},\beta_{N-1})$. Эта точка обладает тем свойством, что расстояние от нее до точки β минимально. Таким образом, для нахождения величины b достаточно взять производную от расстояния по b и приравнять ее к нулю:

$$\frac{d}{db}\left(\left(b-\beta_0\right)^2+\left(b-\beta_1\right)^2+\left(b-\beta_2\right)^2\right)=2\left(\left(b-\beta_0\right)+\left(b-\beta_1\right)+\left(b-\beta_2\right)\right)=$$
$$=3b-\beta_0-\beta_1-\beta_2=0$$

Из этого уравнения находим b и записываем координаты точки β^2:

$$\beta^2=\left(\frac{\beta_0+\beta_1+\beta_2}{3},\frac{\beta_0+\beta_1+\beta_2}{3},\frac{\beta_0+\beta_1+\beta_2}{3},\beta_3,\text{K},\beta_{N-1}\right).$$

Эта процедура продолжается дальше, до тех пор, пока при некотором l не выполнится неравенство $\frac{\sum_{i=0}^{l}\beta_i}{l+1}\geq\beta_{l+1}$ или пока l не окажется равной *N-1*. Оценкой является расстояние от точки β до точки $\beta^l=\left(\frac{\sum_{i=0}^{l}\beta_i}{l+1},\text{K},\frac{\sum_{i=0}^{l}\beta_i}{l+1},\beta_{l+1},\text{K},\beta_{N+1}\right)$. Она равна следующей величине: $H=\sum_{j=0}^{l}\left(\frac{\sum_{i=0}^{l}\beta_i}{l+1}-\beta_j\right)^2$. Производная оценки по выходному сигналу β_m равна:

$$\frac{\partial H}{\partial\beta_m}\begin{cases}\left(\dfrac{\sum_{i=0}^{l}\beta_i}{l+1}-\beta_m\right), & m\leq l,\\ 0, & m>l.\end{cases}$$

Для перехода к производным по исходным выходным сигналам α_i необходимо обратить сделанные на первом этапе вычисления оценки преобразования.

3. Двоичный интерпретатор. Оценка для двоичного интерпретатора строится точно также как и для знакового интерпретатора при кодировании номером канала. Пусть правильным ответом является *k*-ый класс, тогда обозначим через K множество номеров сигналов, которым в двоичном представлении k соответствуют единицы. При уровне надежности оценка задается формулой:

$$H=\sum_{i\notin K}\left(\alpha_i-\left(a+b\right)/2+\varepsilon\right)^2+\sum_{i\in K}\left(\alpha_i-\left(a+b\right)/2-\varepsilon\right)^2.$$

Производная оценки по *i*-му выходному сигналу равна:

$$\frac{\partial H}{\partial\alpha_i}=\begin{cases}2(\alpha_i-(a+b)/2+\varepsilon), i\notin K\\ 2(\alpha_k-(a+b)/2-\varepsilon), i\in K.\end{cases}.$$

4. Порядковый интерпретатор. Для построения оценки по порядковому интерпретатору необходимо предварительно переставить компоненты вектора α в соответствии с подстановкой, кодирующей правильный ответ. Обозначим полученный в результате вектор через β^0. Множество точек, удовлетворяющих условию задачи, описывается системой уравнений $\beta_i^0+\varepsilon\leq\beta_{i+1}^0$, где ε - уровень надежности. Обозначим это множество через ***D.*** Оценка задается расстоянием от точки β до проекции этой точки на множество ***D***. Опишем процедуру вычисления проекции.

1. Просмотрев координаты точки β^0, отметим те номера координат, для которых нарушается неравенство $\beta_i^0+\varepsilon\leq\beta_{i+1}^0$.

2. Множество отмеченных координат либо состоит из одной последовательности последовательных номеров $i, i+1, \text{K}, i+l$, или из нескольких таких последовательностей. Найдем точку β^1, которая являлась бы проекцией точки β^0 на гиперплоскость, определяемую уравнениями $\beta_i^1 + \varepsilon = \beta_{i+1}^1$, где i пробегает множество индексов отмеченных координат. Пусть множество отмеченных координат распадается на n последовательностей, каждая из которых имеет вид

$$\beta^1 = \left(\beta_1^0, \ldots, \beta_{i_1-1}^0, \gamma_1, \gamma_1 + \varepsilon, \ldots, \gamma_1 + l_1\varepsilon, \beta_{i_1+l_1+1}^0, \ldots, \beta_{i_2-1}^0, \gamma_2, \gamma_2 + \varepsilon, \ldots, \gamma_2 + l_2\varepsilon, \ldots, \beta_N^0\right)$$

$\beta_{i_m}, \text{K}, \beta_{i_m+l_m}$, где m - номер последовательности. Тогда точка β^1 имеет вид:

1. Точка β^1 является проекцией, и следовательно, расстояние от β^0 до β^1 должно быть минимальным. Это расстояние равно $\sum_{m=1}^{n}\left[\sum_{j=0}^{l_m}\left(\beta_{i_m+j}^0 - \gamma_m - j\varepsilon\right)^2\right]$. Для нахождения минимума этой функции необходимо приравнять к нулю ее производные по γ_m. Получаем систему уравнений $\sum_{j=0}^{l_m}\left(\beta_{i_m+j}^0 - \gamma_m - j\varepsilon\right) = 0$. Решая ее, находим $\gamma_m = \sum_{j=0}^{l_m}\beta_{i_m+j}^0 \Big/ \left(l_m+1\right) - l_m\varepsilon/2$.
2. Если точка β^1 удовлетворяет неравенствам, приведенным в первом пункте процедуры, то расстояние от нее до точки β^0 является оценкой. В противном случае, повторяем первый шаг процедуры, используя точку β^1 вместо β^0; Объединяем полученный список отмеченных компонентов со списком, полученным при поиске предыдущей точки; находим точку β^2, повторяя все шаги процедуры, начиная со второго.

Отметим, что в ходе процедуры число отмеченных последовательностей соседних индексов не возрастает. Некоторые последовательности могут сливаться, но новые возникать не могут.

После нахождения проекции можно записать оценку:

$$H = \sum_{m=1}^{n}\left[\sum_{j=0}^{l_m}\left(\beta_{i_m+j}^0 - \sum_{j=0}^{l_m}\beta_{i_m+j}^0 \Big/ \left(l_m+1\right) - (l_m - 2j)\varepsilon/2\right)^2\right].$$

Обозначим через I_m m-ую последовательность соседних координат, выделенную при последнем исполнении первого шага процедуры вычисления оценки: $I_m = \left\{i_m, i_m+1, \text{K}, i_m+l_m\right\}$. Тогда производную оценки по выходному сигналу β_i^0 можно записать в следующем виде:

$$\frac{\partial H}{\partial \beta_i} = \begin{cases} \left(\beta_i^0 - \sum_{j=0}^{l_m}\beta_{i_m+j}^0 \Big/ \left(l_m+1\right) - (l_m - 2(i-i_m))\varepsilon/2\right), \exists m: i \in I_m; \\ 0, i \notin \bigcup_{m=1}^{n} I_m. \end{cases}$$

Таким образом, построение оценки по интерпретатору сводится к следующей процедуре.

1. Определяем множество допустимых точек, то есть таких точек в пространстве выходных сигналов, которые интерпретатор ответа будет интерпретировать как правильный ответ со стопроцентным уровнем уверенности.
2. Находим проекцию выданной сетью точки на это множество. Проекцией является ближайшая точка из множества.
3. Записываем оценку как расстояние от точки, выданной сетью, до ее проекции на множество допустимых точек.

6.4 Оценка обучающего множества. Вес примера

В предыдущем разделе был рассмотрен ряд оценок, позволяющих оценить решение сетью конкретного примера. Однако, ситуация, когда сеть хотят обучить решению только одного примера, достаточно редка. Обычно сеть должна научиться решать все примеры обучающего множества. Ряд алгоритмов обучения, которые будут рассматриваться в главе "учитель", требуют возможности обучать сеть решению всех примеров одновременно и, соответственно, оценивать решение сетью всех примеров обучающего множества. Как уже отмечалось, обучение нейронной сети - это процесс минимизации в пространстве обучаемых параметров функции оценки. Большинство алгоритмов обучения используют способность нейронных сетей быстро вычислять вектор градиента функции оценки по обучаемым параметрам. Обозначим оценку отдельного примера через H_i. а оценку всего обучающего множества через H_{OM}. Простейший способ получения H_{OM} из H_i - простая сумма. При этом вектор градиента вычисляется очень просто:

$$H_{OM} = \sum H_i,$$
$$\nabla H_{OM} = \nabla\left(\sum H_i\right) = \sum \nabla H_i.$$

Таким образом, используя способность сети вычислять градиент функции оценки решения одного примера, можно получить градиент функции оценки всего обучающего множества.

Обучение по всему обучающему множеству позволяет задействовать дополнительные механизмы ускорения обучения. Большинство этих механизмов будет рассмотрено в главе ???. В этом разделе будет рассмотрен только один из них - использование весов примеров. Использование весов примеров может быть вызвано одной из следующих причин.

1. Один из примеров плохо обучается.
2. Число примеров разных классов в обучающем множестве сильно отличаются друг от друга.
3. Примеры в обучающем множестве имеют различную достоверность.

Рассмотрим первую причину - пример плохо обучается. Под «плохо обучается» будем понимать медленное снижение оценки данного примера по отношению к снижению оценки по обучающему множеству. Для того чтобы ускорить обучение данного примера, ему можно приписать вес, больший, чем у остальных примеров. При этом оценка по обучающему множеству и ее градиент можно записать в следующем виде: $H_{OM} = \sum w_i H_i; \nabla H_{OM} = \sum w_i \nabla H_i$. где w_i - вес i-го примера. Эту функцию оценки будем называть оценкой взвешенных примеров. При этом градиент, вычисленный по оценке решения сетью этого примера, войдет в суммарный градиент с большим весом, и, следовательно, сильнее повлияет на выбор направления обучения. Этот способ применим также и для коррекции проблем, связанных со второй причиной - разное число примеров разных классов. Однако в этом случае увеличиваются веса всем примерам того класса, в котором меньше примеров. Опыт показывает, что использование весов в таких ситуациях позволяет улучшить обобщающие способности сетей.

В случае различной достоверности примеров в обучающем множестве функция взвешенных примеров не применима. Действительно, если известно, что достоверность ответа в k-ом примере в два раза ниже, чем в l-ом, хотелось бы, чтобы обученная сеть выдавала для k-ого примера в два раза меньший уровень уверенности. Этого можно достичь, если при вычислении оценки k-ого примера будет использоваться в два раза меньший уровень надежности. Оценка обучающего множества в этом случае вычисляется по формуле без весов, а достоверность учитывается непосредственно при вычислении оценки по примеру. Такую оценку будем называть оценкой взвешенной достоверности.

Таким образом, каждый пример может иметь два веса: вес примера и достоверность примера. Кроме того, при решении задач классификации каждый класс может обладать собственным весом. Окончательно функцию оценки по обучающему множеству и ее градиент можно записать в следующем виде:

$$H_{OM}(\varepsilon) = \sum w_i H_i(\delta_i \varepsilon),$$
$$\nabla H_{OM}(\varepsilon) = \sum w_i \nabla H_i.(\delta_i \varepsilon),$$

где w_i - вес примера, δ_i - его достоверность.

6.5 Глобальные и локальные оценки

В предыдущих разделах был рассмотрен ряд оценок. Эти оценки обладают одним общим свойством - для вычисления оценки по примеру, предъявленному сети, достаточно знать выходной вектор, выданный сетью при решении этого примера, и правильный ответ. Такие оценки будем называть локальными. Приведем точное определение.

Определение. Локальной называется любая оценка, являющаяся линейной комбинацией произвольных непрерывно дифференцируемых функций, каждая из которых зависит от оценки только одного примера.

Использование локальных оценок позволяет обучать сеть решению как отдельно взятого примера, так и всего обучающего множества в целом. Однако существуют задачи, для которых невозможно построить локальную оценку. Более того, для некоторых задач нельзя построить даже обучающее множество. Использование нелокальных оценок возможно даже при решении задач классификации.

Приведем два примера нелокальных оценки.

Кинетическая оценка для задачи классификации. Пусть в обучающее множество входят примеры k классов. Требуется обучить сеть так, чтобы в пространстве выходных сигналов множества примеров разных классов были попарно линейно разделимы.

Пусть сеть выдает N выходных сигналов. Для решения задачи достаточно, чтобы в ходе обучения все точки в пространстве выходных сигналов, соответствующие примерам одного класса, собирались вокруг одной точки - центра концентрации класса, и чтобы центры концентрации разных классов были как можно дальше друг от друга. В качестве центра концентрации можно выбрать барицентр множества точек, соответствующих примерам данного класса.

Таким образом, функция оценки должна состоять из двух компонентов: первая реализует притяжение между примерами одного класса и барицентром этого класса, а вторая отвечает за отталкивание барицентров разных классов. Обозначим точку в пространстве выходных сигналов, соответствующую m-му примеру, через α^m, множество примеров i-го класса через I_i, барицентр точек, соответствующих примерам этого класса, через B^i ($B^i = \frac{1}{|I_i|}\sum_{m\in I_i}\alpha^m$), число примеров в i-ом классе через $|B^i|$, а расстояние между точками a и b через $\mathrm{dist}(a,b) = \sum_j \left(a_j - b_j\right)^2$. Используя эти обозначения, можно записать притягивающий компонент функции оценки для всех примеров i-го класса в виде:

$$H_i^P = \sum_{j\in I_i} \mathrm{dist}\left(\alpha^j, B^i\right)$$

Функция оценки H_i^P обеспечивает сильное притяжение для примеров, находящихся далеко от барицентра. Притяжение ослабевает с приближением к барицентру. Компонент функции оценки, отвечающий за отталкивание барицентров разных классов, должен обеспечивать сильное отталкивание близких барицентров и ослабевать с удалением барицентров друг от друга. Такими свойствами обладает гравитационное отталкивание. Используя гравитационное отталкивание можно записать второй компонент функции оценки в виде: $H^O = \sum_{i<j} \mathrm{dist}\left(B^i, B^j\right)^{-1}$. Таким образом, оценку, обеспечивающую сближение точек, соответствующих примерам одного класса, и отталкивание барицентров, можно записать в виде:

$$H = \sum_i H_i^P + H^O = \sum_i \sum_{j\in I_i} \mathrm{dist}\left(\alpha^j, B^i\right) + \sum_{i<j} \mathrm{dist}\left(B^i, B^j\right).$$

Вычислим производную оценки по j-му выходному сигналу, полученному при решении i-го примера. Пусть i-ый пример принадлежит l-му классу. Тогда производная имеет вид:

$$\frac{dH}{d\alpha_j^i} = 2\left(\alpha_j^i - B_j^l\right) - \frac{2}{|I_l|}\sum_{k\neq l}\frac{B_j^l - B_j^k}{\mathrm{dist}^2\left(B^l, B^k\right)}.$$

Эту оценку будем называть кинетической. Существует одно основное отличие этой оценки от всех других, ранее рассмотренных, оценок для решения задач классификации. При использовании традиционных подходов, сначала выбирают интерпретатор ответа, затем строят по выбранному интерпретатору функцию оценки, и только затем приступают к обучению сети. Для кинетической оценки такой

подход не применим. Действительно, до того как будет закончено обучение сети невозможно построить интерпретатор. Кроме того, использование кинетической оценки, делает необходимым обучение сети решению всех примеров обучающего множества одновременно. Это связанно с невозможностью вычислить оценку одного примера. Кинетическая оценка, очевидно, не является локальной: для вычисления производных оценки по выходным сигналам примера необходимо знать барицентры всех классов, для вычисления которых, в свою очередь, необходимо знать выходные сигналы, получаемые при решении *всех* примеров обучающего множества.

Интерпретатор для кинетической оценки строится следующим образом. Для построения разделителя i-го и j-го классов строим плоскость, перпендикулярную к вектору $\left(B^i - B^j\right)$. Уравнение этой плоскости можно записать в виде

$$\frac{\sum_{h=1}^{N}\left(B_h^i - B_p^j\right)\alpha_p}{\sum_{h=1}^{N}\left(B_h^i - B_p^j\right)^2} + D = 0.$$

Для определения константы D находим среди точек i-го класса ближайшую к барицентру j-го класса. Подставляя координаты этой точки в уравнение гиперплоскости, получаем уравнение на D. Решив это уравнение, находим величину D_1. Используя ближайшую к барицентру i-го класса точку j-го класса, находим величину D_2. Искомая константа D находится как среднее арифметическое между D_1 и D_2. Для отнесения произвольного вектора к i-му или j-му классу достаточно подставить его значения в левую часть уравнения разделяющей гиперплоскости. Если значение левой части уравнения получается больше нуля, то вектор относится к j-му классу, в противном случае - к i-му.

Интерпретатор работает следующим образом: если для i-го класса все разделители этого класса с остальными классами выдали ответ i-ый класс, то окончательным ответом является i-ый класс. Если такого класса не нашлось, то ответ «не знаю». Ситуация, когда для двух различных классов все разделители подтвердили принадлежность к этому классу, невозможна, так как разделитель этих двух классов должен был отдать предпочтение одному из них.

Рассмотренный пример решения задачи с использованием нелокальной оценки позволяет выделить основные черты обучения с нелокальной оценкой:

1. Невозможность оценить решение одного примера.
2. Невозможность оценить правильность решения примера до окончания обучения.
3. Невозможность построения интерпретатора ответа до окончания обучения.

Этот пример является отчасти надуманным, поскольку его можно решить с использованием более простых локальных оценок. Ниже приведен пример задачи, которую невозможно решить с использованием локальных оценок.

Генератор случайных чисел. Необходимо обучить сеть генерировать последовательность случайных чисел из диапазона $[0,1]$ с заданными k первыми моментами. Напомним, что для выборки роль первого момента играет среднее значение, второго - средний квадрат, третьего - средний куб и так далее. Есть два пути решения этой задачи. Первый - используя стандартный генератор случайных чисел подготовить задачник и обучить по нему сеть. Этот путь плох тем, что такой генератор будет просто воспроизводить последовательность чисел, записанную в задачнике. Для получения такого результата можно просто хранить задачник.

Второй вариант - обучать сеть без задачника! Пусть нейросеть принимает один входной сигнал и выдает один выходной. При использовании сети выходной сигнал первого срабатывания сети (первое случайное число) будет служить входным сигналом для второго срабатывания сети и так далее.

Для построения оценки зададимся тремя наборами чисел: M_i - необходимое значение i-го момента, L_i - длина последовательности, на которой i-ый момент сгенерированной последовательности должен не более чем на ε_i отличаться от M_i. ε_i - точность вычисления i-го момента.

Выборочная оценка совпадения i-го момента в сгенерированной последовательности на отрезке, начинающемся с j-го случайного числа, вычисляется по следующей формуле: $\overline{M_i^j} = \frac{1}{L_i}\sum_{l=j}^{j+L_i-1}\alpha_l^i$, где α_l - выходной сигнал, полученный на l-ом срабатывании сети. Для оценки точности совпадения i-го

момента в сгенерированной последовательности на отрезке, начинающемся с j-го случайного числа, воспользуемся оценкой числа с допуском ε_i:

$$H_j^i = \begin{cases} 0, & \text{при } \left|\overline{M_i^j} - \mathrm{M}_i\right| \le \varepsilon_i, \\ \left(\overline{M_i^j} - \mathrm{M}_i - \varepsilon_i\right)^2, & \text{при } \overline{M_i^j} > \mathrm{M}_i + , \varepsilon_i \\ \left(\overline{M_i^j} - \mathrm{M}_i + \varepsilon_i\right)^2, & \text{при } \overline{M_i^j} < \mathrm{M}_i - \varepsilon_i. \end{cases}$$

Таким образом, при обучении сети генерации последовательности из N случайных чисел оценку можно записать в следующем виде:

$$H = \sum_{i=1}^{k} \sum_{j=1}^{N-L_i} H_j^i.$$

Производная оценки по выходному сигналу l-го срабатывания сети можно записать в следующем виде:

$$\frac{dH}{d\alpha_l} = \sum_{i=1}^{k} \sum_{j=l-L_i+1}^{l+L_i-1} \begin{cases} 0, & \text{при } \left|\overline{M_i^j} - \mathrm{M}_i\right| \le \varepsilon_i, \\ \dfrac{i\alpha^{i-1}}{L_i}\left(\overline{M_i^j} - \mathrm{M}_i - \varepsilon_i\right), & \text{при } \overline{M_i^j} > \mathrm{M}_i + \varepsilon_i, \\ \dfrac{i\alpha^{i-1}}{L_i}\left(\overline{M_i^j} - \mathrm{M}_i + \varepsilon_i\right), & \text{при } \overline{M_i^j} < \mathrm{M}_i - \varepsilon_i. \end{cases}$$

Используя эту оценку можно обучать сеть генерировать случайные числа. Удобство этого подхода к решению задачи обучения генератора случайных чисел в том, что можно достаточно часто менять инициирующий сеть входной сигнал, что позволит сети генерировать не одну, а много различных последовательностей, обладающих всеми необходимыми свойствами.

При использовании предложенной оценки нет никаких гарантий того, что в генерируемой сетью последовательности не появятся сильно скоррелированные подпоследовательности. Для удаления корреляций можно модифицировать оценку так, чтобы она возрастала при появлении корреляций. Рассмотрим две подпоследовательности длинны L, первая из которых начинается с α_i, а другая с α_{i+h}. Коэффициент корреляции этих последовательностей записывается в виде:

$$r_{ih} = \frac{\dfrac{1}{L}\sum_{j=0}^{L-1} \alpha_{i+j}\alpha_{i+j+h} - \overline{\alpha_i}\,\overline{\alpha_{i+h}}}{\sqrt{\overline{\alpha_i^2} - \overline{\alpha_i}^2}\sqrt{\overline{\alpha_{i+h}^2} - \overline{\alpha_{i+h}}^2}}.$$

В этой формуле приняты следующие обозначения: $\overline{\alpha_i}$ - среднее по последовательности, начинающейся с α_i; $\overline{\alpha_i^2}$ - средний квадрат последовательности начинающейся с α_i. Вычисление такого коэффициента корреляции довольно долгий процесс. Однако вместо выборочных моментов в формулу можно подставить значения моментов, которые последовательность должна иметь. В этом случае формула сильно упрощается:

$$r_{ih} = \frac{\dfrac{1}{L}\sum_{j=0}^{L-1} \alpha_{i+j}\alpha_{i+j+h} - M_1^2}{M_2 - M_1^2}.$$

Добавку для удаления корреляций последовательностей длиной от L_1 до L_2 и смещенных друг относительно друга на смещения от h_1 до h_2 можно записать в виде:

$$H_r = \sum_{L=L_1}^{L_2} \sum_{h=h_1}^{h_2} \sum_{i=1}^{N-h-L+1} \left(\frac{\frac{1}{L} \sum_{j=0}^{L-1} \alpha_{i+j} \alpha_{i+j+h} - M_1^2}{M_2 - M_1^2} \right)^2 .$$

При необходимости можно ввести и другие поправки, учитывающие требования к генератору случайных чисел.

6.6 Составные интерпретатор ответа и оценка

При использовании нейронных сетей для решения различных задач возникает необходимость получать от сети не один ответ, а несколько. Например, при обучении сети решению задачи диагностики отклонений в реакции на стресс нейронная сеть должна была определить наличие или отсутствие тринадцати различных патологий. Если одна сеть может выдавать только один ответ, то для решения задачи необходимо задействовать тринадцать сетей. Однако в этом нет необходимости. Поскольку каждый ответ, который должна выдавать сеть, имеет только два варианта, то можно использовать для его получения классификатор на два класса. Для такого классификатора необходимо два выходных сигнала. Тогда для решения задачи достаточно получать 26 выходных сигналов: первые два сигнала - для определения первой патологии, третий и четвертый - для второй и так далее. Таким образом, интерпретатор ответа для этой задачи состоит из тринадцати интерпретаторов, а оценка из тринадцати оценок. Более того, нет никаких ограничений на типы используемых интерпретаторов или оценок. Возможна комбинация, например, следующих ответов.

1. Число с допуском.
2. Классификатор на восемь классов.
3. Случайное число.

При использовании таких составных оценок и интерпретаторов каждый из этих компонентов должен следить за тем, чтобы каждая частная оценка или интерпретатор получали на вход те данные, которые им необходимы.

6.7 Стандарт первого уровня компонента интерпретатор ответа

Данный раздел посвящен описанию стандарта записи на диск компонента интерпретатор ответов. Построение интерпретатора происходит в редакторе интерпретаторов ответа. Интерпретатор ответа всегда является составным, даже если выходом является один ответ. В состав этого объекта входят частные интерпретаторы. Кроме того, описание интерпретатора должно включать в себя правила распределения выходных сигналов сети между частными интерпретаторами и расположения ответов частных интерпретаторов в едином массиве ответов.

Таким образом, интерпретатор ответа при выполнении запроса на интерпретацию массива выходных сигналов сети получает на входе массив выходных сигналов сети, а возвращает два массива – ответов и коэффициентов уверенности.

Каждый частный интерпретатор ответа получает на входе массив сигналов (возможно из одного элемента), которые он интерпретирует, а на выходе возвращает два числа – ответ и коэффициент уверенности в этом ответе.

Таблица 1.

Ключевые слова языка описания интерпретаторов ответа.

Ключевое слово	Краткое описание
1. Answer	Ответ.
2. Connections	Начало блока описания распределения сигналов и ответов.
3. Contents	Начало блока описания состава интерпретатора.
4. Include	Предшествует имени файла, целиком вставляемого в это место описания.
5. Interpretator	Заголовок раздела файла, содержащий описание интерпретатор.
6. NumberOf	Функция. Возвращает число интерпретируемых частным интерпретатором сигналов.
7. Reliability	Коэффициент уверенности.
8. Signals	Имя, по которому адресуются интерпретируемые сигналы; начало блока описания сигналов.
9. SetParameters	Процедура установления значений параметров.

Таблица 2.

Стандартные частные интерпретаторы.

Название	Параметры	Аргументы	Описание
Empty	B – множитель C – смещение		Интерпретирует один сигнал A. Ответом является величина O=A*B+C
Binary	E – уровень надежности	N – число сигналов (классов)	Кодирование номером канала. Знаковый интерпретатор
Major	E – уровень надежности	N – число сигналов (классов)	Кодирование номером канала. Максимальный интерпретатор.
BynaryCoded	E – уровень надежности	N – число сигналов (классов)	Двоичный интерпретатор.

В табл. 1 приведен список ключевых слов, специфических для языка описания интерпретатора ответов. Наиболее часто встречающиеся интерпретаторы объявлены стандартными. Для стандартных интерпретаторов описание частных интерпретаторов отсутствует. Список стандартных интерпретаторов приведен в табл. 2.

6.7.1 БНФ языка описания интерпретатора

Обозначения, принятые в данном расширении БНФ и описание ряда конструкций приведены в главе «Общий стандарт» в разделе «Описание языка описания компонентов».

<Описание интерпретатора> ::= <Заголовок> [<Описание функций>] <Описание частных интерпретаторов> <Описание состава> [<Установление параметров>] [<Описание сигналов>] [<Описание распределения сигналов>] [<Описание распределения ответов>] <Конец описания интерпретатора>

<Заголовок> ::= **Interpretator** <Имя интерпретатора>

<Имя интерпретатора>::= <Идентификатор>

<Описание частных интерпретаторов> ::= <Описание частного интерпретатора> [<Описание частных интерпретаторов>]

<Описание частного интерпретатора> ::= <Заголовок описания интерпретатора> [<Описание статических переменных >] [<Описание переменных>] <Тело интерпретатора>

<Заголовок описания интерпретатора> ::= **Inter** <Имя частного интерпретатора> : (<Список формальных аргументов>)

<Имя частного интерпретатора> ::= <Идентификатор>

<Тело интерпретатора> ::= **Begin** <Составной оператор> **End**

<Описание состава> ::= **Contents** <Список имен интерпретаторов> ;

<Список имен интерпретаторов> ::= <Имя интерпретатора> [,<Список имен интерпретаторов >]

<Имя интерпретатора> ::= <Псевдоним>: {<Имя ранее описанного интерпретатора> | <Имя стандартного интерпретатора>} [*/*<Число экземпляров >*/*][*(*<Список фактических аргументов>*)*]

<Псевдоним> ::= <Идентификатор>

<Число экземпляров > ::= <Целое число>

<Имя ранее описанного интерпретатора> ::= <Идентификатор>

<Имя стандартного интерпретатора> ::= <Идентификатор>

<Установление параметров> ::= <Установление параметров ***Частного интерпретатора***> [;<Установление параметров>]

<Описание сигналов> ::= **Signals <**Константное выражение типа ***Long>***

<Описание распределения сигналов> ::= <Описание распределения ***Сигналов, Интерпретатора, Частного интерпретатора, Signals***>

<Описание распределения ответов> ::= <Описание распределения ***Ответов, Интерпретатора, Частного интерпретатора, Answer***>

<Конец описания интерпретатора> ::= **End Interpretator**

6.7.2 Описание языка описания интерпретаторов

Структура описания интерпретатора имеет вид: заголовок, описание частных интерпретаторов, описание состава, описание сигналов, описание распределения сигналов, описание распределения ответов, конец описания интерпретатора.

Заголовок состоит из ключевого слова Interpretator и имени интерпретатора и служит для обозначения начала описания интерпретатора в файле, содержащем несколько компонентов нейрокомпьютера.

Описание частного интерпретатора – это описание процедуры, вычисляющей две величины: ответ и уверенность в ответе. Отметим, что уверенность в ответе имеет смысл только для оценок с уровнем надежности. В остальных случаях интерпретатор ответа может вычислять аналогичную величину, но эта величина не является коэффициентом уверенности в ответе в точном смысле. Отметим, что при описании частного интерпретатора его аргументом, как правило, является число интерпретируемых сигналов. При выполнении частный интерпретатор получает в качестве аргументов массив интерпретируемых сигналов и две действительные переменные для возвращения вычисленных ответа и уверенности в ответе. Формально, при исполнении, частный интерпретатор имеет описание следующего вида:
Pascal:

```
Procedure Interpretator(Signals : PRealArray; Var Answer, Reliability : Real);
```

C:

```
void Interpretator(PRealArray Signals, Real* Answer, Real* Reliability);
```

В разделе описания состава перечисляются частные интерпретаторы, входящие в состав интерпретатора. Признаком конца раздела служит символ «;».

В необязательном разделе установления параметров производится задание значений параметров (статических переменных) частных интерпретаторов. После ключевого слова SetParameters следует список значений параметров в том порядке, в каком параметры были объявлены при описании частного интерпретатора (для стандартных интерпретаторов порядок параметров указан в табл. 2). При использовании одного оператора задания параметров для задания параметров нескольким экземплярам одного частного интерпретатора после ключевого слова SetParameters указывается столько выражений, задающих значения параметров, сколько необходимо для одного экземпляра. Например, если в блоке описания состава содержится 10 экземпляров двоичного интерпретатора на 15 интерпретируемых сигналов – MyInt : BinaryCoded(15)[10], то после ключевого слова SetParameters должно быть только одно выражение:

```
MyInt[I:1..10] SetParameters 0.01*I
```

В данном примере первый интерпретатор будет иметь уровень надежности равный 0.01, второй – 0.02 и т.д.

В необязательном разделе описание сигналов указывается число сигналов, интерпретируемых интерпретатором. Если этот раздел опущен, то полагается, что число интерпретируемых интерпретатором сигналов равно сумме сигналов, интерпретируемых всеми частными интерпретаторами. В константном выражении может вызываться функция NumberOf, аргументом которой является имя частного интерпретатора (или его псевдоним) с указанием фактических аргументов.

В необязательном разделе описания распределения сигналов для каждого частного интерпретатора указывается, какие сигналы из общего интерпретируемого массива передаются ему для интерпретации. Если этот раздел отсутствует, то считается, что каждый следующий частный интерпретатор получает следующий фрагмент общего вектора выходных сигналов. В примере 1 данный раздел описывает распределение сигналов по умолчанию.

В необязательном разделе описания распределения ответов для каждого частного интерпретатора указывается, какой элемент массива ответов он вычисляет. Если этот раздел опущен, то считается, что первый частный интерпретатор вычисляет первый элемент массива ответов, второй – второй элемент и т.д. Массив уровней надежностей всегда параллелен массиву ответов. В примере 1 данный раздел описывает распределение ответов по умолчанию.

Кроме того, в любом месте описания интерпретатора могут встречаться комментарии, заключенные в фигурные скобки.

6.7.3 Пример описания интерпретатора

В этом разделе приведены два примера описания одного и того же интерпретатора следующего состава: первый сигнал интерпретируется как температура путем умножения на 10 и добавления 273; следующие два сигнала интерпретируются как наличие облачности, используя знаковый интерпретатор; следующие три сигнала интерпретируются как направление ветра, используя двоичный интерпретатор (восемь румбов); последние три сигнала интерпретируются максимальным интерпретатором как сила осадков (без осадков, слабые осадки, сильные осадки). В первом примере приведено описание дубликатов всех стандартных интерпретаторов. Во втором – использованы стандартные интерпретаторы.

Пример 1.

```
Interpretator Meteorology

Inter Empty1 ()              {Интерпретатор осуществляющий масштабирование и сдвиг сигнала}
   Static
      Real B Name "Масштабный множитель";
```

```
    Real C Name "Сдвиг начала отсчета";
  Begin
    Answer = Signals[1] * B + C;
    Reliability = 0
  End

Inter Binary1 : ( N : Long )                { Кодирование номером канала. Знаковый интерпретатор}
  Static
    Real E Name "Уровень надежности";
  Var
    Long A, B, I;
    Real Dist;
  Begin
    Dist = E;
    B = 0;          {Число единиц}
    A = 0;          {Номер единицы}
    For I = 1 To N Do Begin
      If Abs(Signals[I]) < Dist Then Dist = Abs(Signals[I]);
      If Signals[I] > 0 Then Begin A = I; B = B + 1; End;
    End;
    If B <> 1 Then Answer = 0 Else Answer = A
    Reliability = Abs(Dist / E)
  End

Inter Major1 : ( N : Long)                  { Кодирование номером канала. Максимальный интерпретатор.}
  Static
    Real E Name "Уровень надежности";
  Var
    Real A, B;
    Long I, J;
  Begin
    A = -1.E+30;          {Максимальный сигнал}
    B = -1.E+30;              {Второй по величине сигнал}
    J = 0;                    {Номер максимального сигнала}
    For I = 1 To N Do Begin
      If Signals[I] > A Then Begin B = A; A = Signals[I]; J=I; End
      Else If Signals[I] > B Then B = Signals[I];
    End;
    Answer = J;
    If A - B > E Then Reliability = 1 Else Reliability = (A - B) / E;
  End

Inter BynaryCoded1 : ( N : Long )
  Static
    Real E Name "Уровень надежности";
  Var
    Long A, I;
    Real Dist;
  Begin
    Dist = E;
    A = 0;          {Ответ}
    For I = 1 To N Do Begin
      If Abs(Signals[I]) < Dist Then Dist = Abs(Signals[I]);
      A = A * 2;
      If Signals[I] > 0 Then A = A + 1;
    End;
    Answer = A;
    Reliability = Abs(Dist / E)
  End

Contents Temp : Empty1, Cloud : Binary1(2), Wind : BynaryCoded1(3), Rain : Major1(3);
```

Temp **SetParameters** 10, 273;
Cloud **SetParameters** 0.1;
Wind **SetParameters** 0.2;
Rain **SetParameters** 0.15

Signals NumberOf(**Signals**,Temp) + **NumberOf**(**Signals**, Cloud) + **NumberOf**(**Signals**, Wind) +
NumberOf(**Signals**, Rain)

Connections
Temp.**Signals** <=> **Signals**[1];
Cloud.**Signals**[1..2] <=> **Signals**[2; 3];
Wind.**Signals**[1..3] <=> **Signals**[4..6];
Rain.**Signals**[1..3] <=> **Signals**[7..9]
Temp.**Answer** <=> **Answer**[1];
Cloud.**Answer**[1..2] <=> **Answer**[2];
Wind.**Answer**[1..3] <=> **Answer**[3];
Rain.**Answer**[1..3] <=> **Answer**[4]

End Interpretator

Пример 2.
Interpretator Meteorology

Contents Temp : Empty, Cloud : Binary(2), Wind : BynaryCoded(3), Rain : Major(3);

Temp **SetParameters** 10, 273;
Cloud **SetParameters** 0.1;
Wind **SetParameters** 0.2;
Rain **SetParameters** 0.15

End Interpretator

6.8 Стандарт второго уровня компонента интерпретатор ответа

Запросы к компоненту интерпретатор ответа можно разбить на пять групп:

1. Интерпретация.
2. Изменение параметров.
3. Работа со структурой.
4. Инициация редактора и конструктора интерпретатора ответа.
5. Обработка ошибок.

Поскольку нейрокомпьютер может работать одновременно с несколькими сетями, то и компонент интерпретатор ответа должен иметь возможность одновременной работы с несколькими интерпретаторами. Поэтому большинство запросов к интерпретатору содержат явное указание имени интерпретатора ответа. Ниже приведено описание всех запросов к компоненту интерпретатор ответа. Каждый запрос является логической функцией, возвращающей значение истина, если запрос выполнен успешно, и ложь – при ошибочном завершении исполнения запроса.

В запросах второй и третьей группы при обращении к частным интерпретаторам используется следующий синтаксис:

<Полное имя частного интерпретатора> ::=
<Имя интерпретатора>.<Псевдоним частного интерпретатора> [/<Номер экземпляра>/]

Таблица 3.

Значения предопределенных констант компонента интерпретатор ответа и оценка

Название	Величина	Значение
Empty	0	Интерпретирует один сигнал как действительное число.
Binary	1	Кодирование номером канала. Знаковый интерпретатор
Major	2	Кодирование номером канала. Максимальный интерпретатор.
BynaryCoded	3	Двоичный интерпретатор.
UserType	-1	Интерпретатор, определенный пользователем.

При вызове ряда запросов используются предопределенные константы. Их значения приведены в табл. 3.

6.8.1 Запрос на интерпретацию

Единственный запрос первой группы выполняет основную функцию компонента интерпретатор ответа – интерпретирует массив сигналов.

6.8.1.1 Интерпретировать массив сигналов (Interpretate)

Описание запроса:

Pascal:

```
Function Interpretate( IntName : PString; Signals : PRealArray;
    Var Reliability, Answers : PRealArray ) : Logic;
```

C:

```
Logic Interpretate(PString IntName, PRealArray Signals, PRealArray* Reliability, PRealArray* Answers)
```

Описание аргумента:

IntName – указатель на строку символов, содержащую имя интерпретатора ответа.

Signals – массив интерпретируемых сигналов.

Answers – массив ответов.

Reliability – массив коэффициентов уверенности в ответе.

Назначение – интерпретирует массив сигналов Signals, используя интерпретатор ответа, указанный в параметре IntName.

Описание исполнения.

1. Если Error <> 0, то выполнение запроса прекращается.
2. Если в качестве аргумента IntName дан пустой указатель, или указатель на пустую строку, то исполняющим запрос объектом является первый интерпретатор ответа в списке интерпретаторов компонента интерпретатор.
3. Если список интерпретаторов компонента интерпретатор пуст или имя интерпретатора ответа, переданное в аргументе IntName в этом списке не найдено, то возникает ошибка 501 – неверное имя интерпретатора ответа, управление передается обработчику ошибок, а обработка запроса прекращается.
4. Производится интерпретация ответа интерпретатором ответа, имя которого было указано в аргументе IntName.
5. Если во время выполнения запроса возникает ошибка, то генерируется внутренняя ошибка 504 - ошибка интерпретации. Управление передается обработчику ошибок. Выполнение запроса прекращается. В противном случае выполнение запроса успешно завершается.

6.8.2 Остальные запросы

Ниже приведен список запросов, исполнение которых описано в главе "Общий стандарт":

aiSetCurrent – Сделать интерпретатор ответа текущим
aiAdd – Добавление нового интерпретатора ответа
aiDelete – Удаление интерпретатора ответа
aiWrite – Запись интерпретатора ответа
aiGetStructNames – Вернуть имена частных интерпретаторов
aiGetType – Вернуть тип частного интерпретатора
aiGetData – Получить параметры частного интерпретатора
aiGetName – Получить имена параметров частного интерпретатора
aiSetData – Установить параметры частного интерпретатора
aiEdit – Редактировать интерпретатор ответа
OnError – Установить обработчик ошибок
GetError – Дать номер ошибки
FreeMemory – Освободить память

В запросе aiGetType в переменной TypeId возвращается значение одной из предопределенных констант, перечисленных в табл. 3.

При исполнении запроса aiSetData генерируется запрос SetEstIntParameters к компоненте оценка. Аргументы генерируемого запроса совпадают с аргументами исполняемого запроса

6.8.3 Ошибки компонента интерпретатор ответа

В табл. 4 приведен полный список ошибок, которые могут возникать при выполнении запросов компонентом интерпретатор ответа, и действия стандартного обработчика ошибок.

Таблица 4.

Ошибки компонента интерпретатор ответа и действия стандартного обработчика ошибок.

№	Название ошибки	Стандартная обработка
501	Неверное имя интерпретатора ответа	Занесение номера в Error
502	Ошибка считывания интерпретатора ответа	Занесение номера в Error
503	Ошибка сохранения интерпретатора ответа	Занесение номера в Error
504	Ошибка интерпретации	Занесение номера в Error

6.9 Стандарт первого уровня компонента оценка

Данный раздел посвящен описанию стандарта хранения на диске описания компонента оценка. Построение оценки происходит в редакторе оценок. В данном стандарте предлагается ограничиться рассмотрением только локальных оценок, поскольку использование нелокальных (глобальных) оценок сильно усложняет компонент оценка, а область применения нелокальных оценок узка по сравнению с локальными оценками.

Оценка всегда является составной, даже если ответом сети является одна величина. В состав этого объекта входят частные оценки. Кроме того, в описание оценки включаются правила распределения выходных сигналов сети между частными оценками и расположения оценок, вычисляемых частными оценками, в едином массиве оценок. Кроме того, различные частные оценки могут иметь разную значимость. В этом случае общая оценка определяется как сумма частных оценок с весами, задающими значимость.

Таким образом, оценка при выполнении запроса на оценивание массива выходных сигналов сети получает на входе массив выходных сигналов сети, массив правильных ответов и массив их достоверностей, а возвращает два массива – массив оценок и массив производных оценки по выходным сигналам сети – и величину суммарной оценки. Возможны два режима оценивания: оценивание без вычисления массива производных оценки по выходным сигналам сети, и оценивание с вычислением массива производных.

Таблица 5

Ключевые слова языка описания оценок.

Ключевое слово	Краткое описание
1. Answer	Правильный ответ.
2. Back	Массив производных оценки по оцениваемым сигналам.
3. Contents	Начало блока описания состава оценки.
4. Direv	Признак необходимости вычисления производных.
5. Est	Заголовок описания частной оценки.
6. Estim	Переменная действительного типа, для возвращения вычисленной оценки.
7. Estimation	Заголовок раздела файла, содержащий описание оценки.
8. Include	Предшествует имени файла, целиком вставляемого в это место описания.
9. Link	Указывает интерпретатор ответа, связанный с оценкой.
10. NumberOf	Функция. Возвращает число интерпретируемых частным интерпретатором сигналов.
11. Reliability	Достоверность правильного ответа.
12. Signals	Имя, по которому адресуются интерпретируемые сигналы; начало блока описания сигналов.
13. Weight	Вес частной оценки.
14. Weights	Начало блока описания весов частных оценок.

Каждая частная оценка получает на входе свой массив сигналов (возможно из одного элемента), правильный ответ и его достоверность, а на выходе вычисляет оценку и, при необходимости, массив производных оценки по выходным сигналам сети.

В табл. 5 приведен список ключевых слов специфических для языка описания оценок. Наиболее часто встречающиеся частные оценки объявлены стандартными. Для стандартных оценок описание частных оценок отсутствует. Список стандартных оценок приведен в табл. 6.

Таблица 6

Стандартные частные оценки.

Название	Параметры	Аргументы	Описание
Empty	B – множитель C – смещение		Оценивает один сигнал A, вычисляя расстояние до правильного ответа с учетом нормировки.
Binary	E – уровень надежности	N – число сигналов.	Кодирование номером канала. Соответствует знаковому интерпретатору.
Major	E – уровень надежности	N – число сигналов.	Кодирование номером канала. Соответствует максимальному интерпретатору.
BynaryCoded	E – уровень надежности	N – число сигналов.	Соответствует двоичному интерпретатору.

6.9.1 БНФ языка описания оценок

Обозначения, принятые в данном расширении БНФ и описание ряда конструкций приведены в главе «Общий стандарт» в разделе «Описание языка описания компонент».

<Описание оценки> ::= <Заголовок> [<Описание функций>] <Описание частных оценок> <Описание состава> [<Связывание с интерпретаторами>] [<Установление параметров>] [<Описание весов>] [<Описание сигналов>] [<Описание распределения сигналов>] [<Описание распределения оценок>] <Конец описания оценки>

<Заголовок> ::= **Estimation** <Имя оценки>

<Имя оценки>::= <Идентификатор>

<Описание частных оценок> ::= <Описание частной оценки> [<Описание частных оценок>]

<Описание частной оценки> ::= <Заголовок описания оценки> [<Описание статических переменных>] [<Описание переменных>] <Тело оценки>

<Заголовок описания оценки> ::= **Est** <Имя частной оценки> (<Список формальных аргументов>)

<Имя частной оценки> ::= <Идентификатор>

<Тело оценки> ::= **Begin** <Составной оператор> **End**

<Описание состава> ::= **Contents** <Список имен оценок> ;

<Список имен оценок> ::= <Имя оценки> [,<Список имен оценок >]

<Имя оценки> ::= <Псевдоним>: {<Имя ранее описанной оценки> | <Имя стандартной оценки>} [*(*<Список фактических аргументов>*)*] [*[*<Число экземпляров>*]*]

<Псевдоним> ::= <Идентификатор>

<Число экземпляров > ::= <Целое число>

<Имя ранее описанной оценки> ::= <Идентификатор>

<Имя стандартной оценки> ::= <Идентификатор>

<Установление параметров> ::= <Установление параметров ***Частной оценки***> [;<Установление параметров>]

<Связывание с интерпретаторами> ::= <Псевдоним> [*[*<Начальный номер> [..<Конечный номер> [:<Шаг>]]*]*] **Link** <Псевдоним интерпретатора> [*[*<Начальный номер> [..<Конечный номер> [:<Шаг>]]*]*]

<Псевдоним интерпретатора> ::= <Идентификатор>

<Описание весов> ::= **Weights** <Список весов>;

<Список весов> ::= <Вес> [,<Список весов>]

<Вес> ::= <Действительное число>

<Описание сигналов> ::= **Signals <**Константное выражение типа ***Long>***

<Описание распределения сигналов> ::= <Описание распределения ***Сигналов, Оценки, Частной оценки, Signals***>

<Описание распределения ответов> ::= <Описание распределения ***Ответов, Оценки, Частной оценки, Answer***>

<Конец описания оценки> ::= **End Estimation**

6.9.2 Описание языка описания оценок

Структура описания оценки имеет вид: заголовок, описание функций, описание частных оценок, описание состава, описание связей с интерпретаторами, описание сигналов, описание распределения сигналов, описание распределения ответов, конец описания оценки.

Заголовок состоит из ключевого слова Estimation и имени оценки и служит для обозначения начала описания оценки в файле, содержащем несколько компонент нейрокомпьютера.

Описание частной оценки – это описание процедуры, вычисляющей оценку и, при необходимости, массив производных оценки по выходным сигналам сети. Отметим, что при описании частной оценки его аргументом, как правило, является число оцениваемых сигналов. При выполнении частная оценка получает в качестве аргументов массив оцениваемых сигналов, признак необходимости вычисления производных, правильный ответ, достоверность правильного ответа, действительную переменную для возвращения вычисленной оценки и массив для возвращения производных. Формально, при исполнении частная оценка имеет описание следующего вида:

Pascal:

```
Procedure Estimation(Signals, Back : PRealArray; Direv : Logic; Answer, Reliability : Real; Var Estim : Real);
```

C:

```
void Estimation(PRealArray Signals, PRealArray Back,
Logic Direv, Real Answer, Real Reliability, Real* Estim);
```

Отметим одну важную особенность выполнения тела частной оценки. Оператор присваивания значения элементу массива производных, означает добавление этого значения к величине, ранее находившейся в этом массиве. Например, запись **Back**[I] = A, означает выполнение следующего оператора **Back**[I] = **Back**[I] + A. Это связано с тем, что один и тот же сигнал может быть задействован в нескольких частных оценках и производная общей функции оценки равна сумме производных частных оценок по этому сигналу.

В разделе описания состава перечисляются частные оценки, входящие в состав оценки. Признаком конца раздела служит символ «;».

В необязательном разделе установления параметров производится задание значений параметров частных оценок. После ключевого слова SetParameters следует список значений параметров в том порядке, в каком параметры (статические переменные) были объявлены при описании частной оценки (для стандартных оценок порядок параметров указан в табл. 6). При использовании одного оператора задания параметров для задания параметров нескольким экземплярам одной частной оценки после ключевого слова SetParameters указывается столько выражений, задающих значения параметров, сколько необходимо для одного экземпляра. Например, если в блоке описания состава содержится 10 экземпляров двоичной оценки на 15 оцениваемых сигналов – MyEst : BinaryCoded(15)[10], то после ключевого слова SetParameters должно быть только одно выражение:

```
MyEst[I:1..10] SetParameters 0.01*I
```

В данном примере первая оценка будет иметь уровень надежности равный 0.01, вторая – 0.02 и т.д.

В необязательном разделе описания связей с интерпретаторами можно указать интерпретатор ответа, связанный с данной оценкой. Для связи интерпретатор и оценка должны иметь одинаковое число параметров и одинаковый порядок их описания. Так, в приведенном ниже примере, невозможно связывание оценки Temp с одноименным интерпретатором из-за различия в числе параметров. Если в левой части выражения Link указан диапазон оценок, то в правой части должен быть указан диапазон, содержащий столько же интерпретаторов. Указание связи влечет идентичность параметров оценки и связанного с ней интерпретатора ответов. Идентичность обеспечивается при исполнении запросов aiSetData и esSetData.

В необязательном разделе описания весов указываются веса частных оценок. Если этот раздел опущен, то все частные оценки равны единице, то есть все частные оценки имеют равную значимость.

В необязательном разделе описания сигналов указывается число сигналов, оцениваемых всеми частными оценками. Если этот раздел опущен, то полагается, что число оцениваемых оценкой сигналов равно сумме сигналов, оцениваемых всеми частными оценками. В константном выражении может вызываться функция NumberOf, аргументом которой является имя частной оценки (или ее псевдоним) с указанием фактических аргументов.

В необязательном разделе описания распределения сигналов для каждой частной оценки указывается, какие сигналы из общего оцениваемого массива передаются ей для оценивания. Если этот раздел опущен, то считается, что каждая следующая частная оценка получает следующий фрагмент массива сигналов. Порядок следования частных оценок соответствует порядку их перечисления в разделе описания состава. В примере 1 раздел описания распределения сигналов задает распределение сигналов по умолчанию. Массив производных оценки по выходным сигналам сети параллелен массиву сигналов.

В необязательном разделе описания распределения ответов для каждой частной оценки указывается какой элемент массива ответов будет ей передан. Если этот раздел опущен, то считается, что каждая следующая частная оценка получает следующий элемент массива ответов. Порядок следования частных оценок соответствует порядку их перечисления в разделе описания состава. В примере 1 раздел описания распределения ответов задает распределение сигналов по умолчанию. Массивы достоверностей ответов и вычисленных оценок параллельны массиву ответов.

Кроме того, в любом месте описания оценки могут встречаться комментарии, заключенные в фигурные скобки.

6.9.3 Пример описания оценки

В этом разделе приведены два примера описания одной и той же оценки следующего состава: первый сигнал интерпретируется как температура путем умножения на 10 и добавления 273; следующие два сигнала интерпретируются как наличие облачности, используя знаковый интерпретатор; следующие три сигнала интерпретируются как направление ветра, используя двоичный интерпретатор (восемь румбов); последние три сигнала интерпретируются максимальным интерпретатором как сила осадков (без осадков, слабые осадки, сильные осадки). Для трех последних интерпретаторов используются соответствующие им оценки типа расстояние до множества. В первом примере приведено описание дубликатов всех стандартных оценок. Во втором – использованы стандартные оценки.

Пример 1.

```
Estimation Meteorology

Est Empty1 ()     {Оценка для интерпретатора, осуществляющего масштабирование и сдвиг сигнала}
   Static
      Real B Name "Масштабный множитель";
      Real C Name "Сдвиг начала отсчета";
      Real E Name "Требуемая точность совпадения";
   Var
      Real A;
   Begin
      A = Signals[1] - (Answer - C) / B;
      D = E * Reliability;      {Уровень надежности с поправкой на достоверность}
      If Abs(A)<D Then Estim = 0
      Else
         If A > 0 Then Begin
            Estim = Weight * Sqr(A - D) / 2;
            If Direv Then Back[1] = Weight * (A - D);
         End Else Begin
            Estim = Weight * Sqr(A + D) / 2;
            If Direv Then Back[1] = Weight * (A + D);
         End
   End

Est Binary1 ( N : Long)      { Кодирование номером канала. Оценка для знакового интерпретатора.}
   Static
      Real E Name "Уровень надежности;
   Var
      Long I, J;
      Real A, B, C;
   Begin
      J = Answer;               {Правильный ответ – номер правильного класса}
      B = 0;
      C = E * Reliability;      {Уровень надежности с поправкой на достоверность}
      For I = 1 To N Do
         If  I = J Then Begin
            If Signals[I] < C Then Begin
               B = B + Sqr(Signals[I] - C);
               If Direv Then Back[I] = 2 * Weight * (Signals[I]-C);
            End;
         End Else Begin
            If Signals[I] > -C Then Begin
               B = B + Sqr(Signals[I] + C);
               If Direv Then Back[I] = 2 * Weight * (Signals[I] + C);
            End
         End;
      Estim = Weight*B
   End
```

```
Est Major1 ( N : Long) {Кодирование номером канала. Оценка для максимального интерпретатора.}
    Static
        Real E Name "Уровень надежности;
    Var
        Real A, B;
        Long I, J, K, Ans;
        RealArray[N+1] Al,Ind;
    Begin
        Ans = Answer;
        Ind[1] = Ans;
        Al[1] = Signals[Ans] - E * Reliability;
        Ind[N+1] = 0;
        Al[N+1] = -1.e40;
        K:=1;
        For I = 1 To N Do
            If I <> Ans Then Begin
                Al[K] = Signals[I];
                Ind[K] = I;
                K = K + 1;
             End;                          {Подготовлен массив сигналов}
        For I = 2 To N-1 Do Begin
            A = Al[I];
            K = I;
            For J = I+1 To N Do
                If Al[J] > A Then Begin
                    K = J;
                    A = Al[J];
                End;                       {Найден следующий по величине}
            Al[K] = Al[I];
            Al[I] = A;
            J = Ind[K];
            Ind[K] = Ind[I];
            Ind[I] = J;
        End;                               {Массивы отсортированы}
        A = Al[1];                        {Сумма первых I членов}
        I = 1;
        While (A / I <= Al[I+1]) Do Begin
            A = A + Al[I];
            I = I + 1;
        End;                               {В конце цикла I-1 равно числу корректируемых сигналов}
        B = A / I;                         {B – величина, к которой должны стремиться}
        A = 0;                             {корректируемые сигналы}
        For J = 1 To I Do Begin
            A = A + Sqr(Al[J] - B);
            If Direv Then Back[Ind[J]] = -2* Weight * (Al[J] - B);
        End;
        Estim = Weight * A
    End;

Est BynaryCoded1 : ( N : Long)             {Оценка для кодирования номером канала}
    Static
        Real E Name "Уровень надежности;
    Var
        Long I, J, A, K;
        Real B, C;
    Begin
        A = Answer;
        B = 0;
        C = E * Reliability;               {Уровень надежности с поправкой на достоверность}
```

```
        For I = N To 1 By -1 Do Begin
            J = A / 2;
            K = A - 2 * J;
            A = J;
            If  A = 1 Then Begin
                If Signals[I] < C Then Begin
                    B = B + Sqr(Signals[I] - C);
                    If Direv Then Back[I] = 2 * Weight * (Signals[I]-C);
                End;
            End Else Begin
                If Signals[I] > -C Then Begin
                    B = B + Sqr(Signals[I] + C);
                    If Direv Then Back[I] = 2 * Weight * (Signals[I] + C);
                End;
            End;
        Estim = Weight*B
    End

Contents Temp : Empty1, Cloud : Binary1(2), Wind : BynaryCoded1(3), Rain : Major1(3);

Cloud Link Meteorology.Cloud            {Связываем оценки с интерпретаторами}
Wind Link Meteorology.Wind
Rain Link Meteorology.Rain

Temp SetParameters 10, 273;             {Устанавливаем значения параметров оценок}
Cloud SetParameters 0.1;                {и интерпретаторов}
Wind SetParameters 0.2;
Rain SetParameters 0.15

Weights 1, 1, 1, 1

Signals NumberOf(Signals,Temp) + NumberOf(Signals, Cloud) + NumberOf(Signals, Wind) +
        NumberOf(Signals, Rain)

Connections
    Temp.Signals <=> Signals[1];
    Cloud.Signals[1..2] <=> Signals[2; 3];
    Wind.Signals[1..3] <=> Signals[4..6];
    Rain.Signals[1..3] <=> Signals[7..9]
    Temp.Answer <=> Answer[1];
    Cloud.Answer[1..2] <=> Answer[2];
    Wind.Answer[1..3] <=> Answer[3];
    Rain.Answer[1..3] <=> Answer[4]

End Interpretator
```

Пример 2.

```
Estimation Meteorology

Contents Temp : Empty, Cloud : Binary(2), Wind : BynaryCoded(3), Rain : Major(3);

Cloud Link Meteorology.Cloud            {Связываем оценки с интерпретаторами}
Wind Link Meteorology.Wind
Rain Link Meteorology.Rain

Temp SetParameters 10, 273;             {Устанавливаем значения параметров оценок}
Cloud SetParameters 0.1;                {и интерпретаторов}
Wind SetParameters 0.2;
Rain SetParameters 0.15

End Interpretator
```

6.10 Стандарт второго уровня компонента оценка

Запросы к компоненте оценка можно разбить на пять групп:

1. Оценивание.
2. Изменение параметров.
3. Работа со структурой.
4. Инициация редактора и конструктора оценки.
5. Обработка ошибок.

Поскольку нейрокомпьютер может работать одновременно с несколькими сетями, то и компонент оценка должен иметь возможность одновременной работы с несколькими оценками. Поэтому большинство запросов к оценке содержат явное указание имени оценки. Ниже приведено описание всех запросов к компоненту оценка. Каждый запрос является логической функцией, возвращающей значение истина, если запрос выполнен успешно, и ложь – при ошибочном завершении исполнения запроса.

В запросах второй и третьей группы при обращении к частным оценкам используется следующий синтаксис:

<Полное имя частной оценки> ::=
<Имя оценки>.<Псевдоним частной оценки> [/<Номер экземпляра>/]

При вызове ряда запросов используются предопределенные константы. Их значения приведены в табл. 3.

6.10.1 Запрос на оценивание

Единственный запрос первой группы выполняет основную функцию компонента оценка – вычисляет оценку и, если требуется, массив производных оценки по оцениваемым сигналам.

6.10.1.1 Оценить массив сигналов (Estimate)

Описание запроса:

Pascal:

Function Estimate(EstName : PString; Signals, Back, Answers, Reliability: PRealArray;
Direv : Logic; Var Estim : Real) : Logic;

C:

Logic Estimate(PString EstName, PRealArray Signals, PRealArray* Back, PRealArray Answers,
PRealArray Reliability, Logic Direv, Real* Estim)

Описание аргумента:

EstName – указатель на строку символов, содержащую имя оценки.
Signals – указатель на массив оцениваемых сигналов.
Back – указатель на массив производных оценки по оцениваемым сигналам.
Answers – указатель на массив правильных ответов.
Reliability – указатель на массив достоверностей правильных ответов.
Direv – признак необходимости вычисления производных (False – не вычислять).
Estim – вычисленная оценка.

Назначение – вычисляет оценку массива сигналов Signals, используя оценку, указанную в параметре EstName.

Описание исполнения.

1. Если Error <> 0, то выполнение запроса прекращается.
2. Если в качестве аргумента EstName дан пустой указатель, или указатель на пустую строку, то исполняющим запрос объектом является первая оценка в списке оценок компонента оценка.
3. Если список оценок компонента оценка пуст или имя оценки, переданное в аргументе EstName, в этом списке не найдено, то возникает ошибка 401 – неверное имя оценки, управление передается обработчику ошибок, а обработка запроса прекращается.
4. Производится вычисление оценки оценкой, имя которой было указано в аргументе EstName.
5. Если во время выполнения запроса возникает ошибка, то генерируется внутренняя ошибка 404 – ошибка оценивания. Управление передается обработчику ошибок. Выполнение запроса прекращается. В противном случае выполнение запроса успешно завершается.

6.10.2 Остальные запросы

Ниже приведен список запросов, исполнение которых описано в главе "Общий стандарт":

esSetCurrent – Сделать оценку текущим
esAdd – Добавление новой оценки
esDelete – Удаление оценки
esWrite – Запись оценки
esGetStructNames – Вернуть имена частных оценок

esGetType – Вернуть тип частной оценки
esGetData – Получить параметры частной оценки
esGetName – Получить имена параметров частной оценки
esSetData – Установить параметры частной оценки
esEdit – Редактировать оценку
OnError – Установить обработчик ошибок
GetError – Дать номер ошибки
FreeMemory – Освободить память

В запросе esGetType в переменной TypeId возвращается значение одной из предопределенных констант, перечисленных в табл. 3.

Кроме того, во второй группе запросов есть запрос SetEstIntParameters аналогичный запросу esSetData, но определяющий частную оценку, параметры которой изменяются, по полному имени связанного с ней интерпретатора ответа.

6.10.2.1 Установить параметры (SetEstIntParameters)

Описание запроса:

Pascal:

```
Function SetEstIntParameters( IntName : PString; Param : PRealArray ) : Logic;
```

C:

```
Logic SetEstIntParameters(PString IntName, PRealArray Param)
```

Описание аргументов:

IntName – указатель на строку символов, содержащую полное имя частного интерпретатора ответа.

Param – адрес массива параметров.

Назначение – заменяет значения параметров частной оценки, связанной с интерпретатором ответа, указанного в аргументе IntName, на значения, переданные, в аргументе Param.

Описание исполнения.

1. Запрос передается всем частным оценкам всех оценок в списке оценок компонента оценка.
2. Если частная оценка связана с частным интерпретатором ответа, имя которого указано в аргументе IntName, то текущие значения параметров частной оценки заменяются на значения, хранящиеся в массиве, адрес которого передан в аргументе Param,.

6.10.3 Ошибки компонента оценка

В табл. 7 приведен полный список ошибок, которые могут возникать при выполнении запросов компонентом оценка, и действия стандартного обработчика ошибок.

Таблица 7.

Ошибки компонента оценка и действия стандартного обработчика ошибок.

№	Название ошибки	Стандартная обработка
401	Неверное имя оценки	Занесение номера в Error
402	Ошибка считывания оценки	Занесение номера в Error
403	Ошибка сохранения оценки	Занесение номера в Error
404	Ошибка вычисления оценки	Занесение номера в Error