Romanuke V.V.
Khmelnitskiy National University,
Khmelnitskiy, Ukraine
E-mail: romanukevadimv@gmail.com

A METHOD OF RESUME-TRAINING OF DISCONTINUOUS WEAR STATE TRACKERS FOR COMPOSING BOOSTING HIGH-ACCURATE ENSEMBLES NEEDED TO REGARD STATISTICAL DATA INACCURACIES AND SHIFTS

UDC 539.375.6+539.538+519.237.8

For tracking metal wear states at bad statistical data inaccuracies and shifts, there is a method of resume-training of discontinuous wear state trackers for boosting them within high-accurate ensembles. These trackers are Gaussian-noised-data-trained two-layer perceptrons. An ordinary tracker is selected and, if its performance is satisfactory, it is resumed-trained cyclically. Number of additional passes of training sets is limited. The resume-training procedure wholly can be cycled.

Key words: metal wear, wear state tracker, statistical data inaccuracies and shifts, boosting high-accurate ensemble.

Use of discontinuous wear state trackers

Particularly, discontinuous wear state trackers (DWST) allow to watch and control metal wear states (MWS), whose range is sampled into a scale starting from the initial wears and ending up to the ultimate wears. DWST get a set of wear influencing factors (WIF), including direct (speed, pressure, temperature, etc.) and indirect (time duration) ones, and return the wear state number (WSN). According to WSN, the controller assigns an operating mode, load, pressure, torque, cooldown, if any. If WSN is returned inaccurate, its aftermath influences badly on both the processed metal billets and metal tools. In their compositions, DWST are boosted up [1, 2] to a high-accurate DWST ensemble (HADWSTE) capable to track MWS at bad statistical data inaccuracies and shifts (SDIS). Any other compositions of wear predictors issuing from stochastic differential equations track worse as wear increases. For composing HADWSTE, however, DWST of perfected accuracy are required as well.

Composing HADWSTE for tracking MWS at higher intensities of SDIS

Two-layer perceptron with nonlinear transfer functions (2LPNLTF) is a universal statistical approximator, fitting to track MWS [1]. For its identification, it requires finite statistical data set (FSDS) $F_L = \left\{ \mathbf{X}_j,\, w_j \right\}_{j=1}^{L}$ including each of $N \in \Gamma \setminus \{1\}$ wear states by $L \in \Gamma \setminus \left\{ \overline{1,\, N-1} \right\}$ and the j-th WSN $w_j \in \left\{ \overline{1,\, N} \right\}$ for $Q \in \Gamma$ WIF within the point $\mathbf{X}_j = \left[x_i^{\langle j \rangle} \right]_{Q \times 1} \in X$. FSDS is accumulated via assigning the fixed original groups of WIF to WSN. These groups should be quite different. And if $L > N$ then there are similar WIF groups, among which unique groups are selected and assigned to their corresponding classes. Non-assigned groups are to be sent into training sets. These sets are

$$\left\{ \mathbf{Y} = \left[y_{is} \right]_{Q \times N} : y_{is} = x_i^{\langle j_s \rangle} \right\} \quad \text{and} \quad Y_R^{(H)} := \left\{ \left\langle \left\{ Y \right\}_{r=1}^{R}, \left\{ \tilde{Y}_h \right\}_{h=1}^{H} \right\rangle : \tilde{Y}_h := Y + \sigma_h \Xi + \mu \sigma_h \Theta, \sigma_h := h \sigma_0 H^{-1} \forall\, h := \overline{1, H}, H \in \Gamma, \sigma_0 > 0, \Xi := \left[\xi_{is} \right]_{Q \times N}, \xi_{is} \in \mathbf{N}(0,1), \mu > 0, \Theta := \left[\theta_{is} \right]_{Q \times N}, \theta_{is} := \zeta_s \in \mathbf{N}(0,1) \right\}$$

$$Y_R^{(H)} := \left\{ \left\langle \left\{ Y \right\}_{r=1}^{R}, \left\{ \tilde{Y}_h \right\}_{h=1}^{H} \right\rangle : \tilde{Y}_h := Y + \sigma_h \Xi + \mu \sigma_h \Theta, \sigma_h := h \sigma_0 H^{-1} \forall\, h := \overline{1, H}, H \in \Gamma, \sigma_0 > 0, \Xi := \left[\xi_{is} \right]_{Q \times N}, \xi_{is} \in \mathbf{N}(0,1), \mu > 0, \Theta := \left[\theta_{is} \right]_{Q \times N}, \theta_{is} := \zeta_s \in \mathbf{N}(0,1) \right\} \quad (1)$$

for WIF original groups and SDIS correspondingly, where $\left\{ x_i^{\langle j_s \rangle} \in [0;\, 1] \right\}_{i=1}^{Q}$ and $R \in \Gamma \cup \{0\}$, and $\mathbf{N}(0,\, 1)$ is the infinite set of standard normal variate's values. In statement (1), the term $\sigma_h \Xi$ models jitter inaccuracies and omissions in statistical data or measurements, and the term $\mu \sigma_h \Theta$ models WIF shifts in every state. Coefficient σ_0 characterizes ultimate jitters and μ is ratio between WIF shifts and the suspected jitters [2].

For tracking MWS at higher intensities of SDIS, HADWSTE is composed whose output is

$$\tilde{s}_* \in \arg\max_{s=\overline{1,\, N}} \tilde{m}_s \quad \text{by} \quad \tilde{m}_s = \sum_{\alpha_*=1}^{B} \lambda(\alpha_*)\, m_s(\alpha_*) \quad \text{at} \quad \lambda(\alpha_*) \ldots 0 \quad \forall\, \alpha_* = \overline{1, B} \quad \text{and} \quad \sum_{\alpha_*=1}^{B} \lambda(\alpha_*) = 1 \quad (2)$$

by the α_*-th 2LPNLTF [2], giving the value $m_s(\alpha_*)$ in its s-th output neuron weighted with $\lambda(\alpha_*)$, $B \in \Gamma \setminus \{1\}$. However, only high-accurate 2LPNLTF (HA2LPNLTF) are required for that. For example, for a problem of tracking 24 MWS with 16 WIF in [2], it had taken 17818 ordinary Gaussian-noised-data-trained

(GND-trained) 2LPNLTF by (1) with 70 hidden layer neurons by $R = 1$ and $H = 18$ for getting 60 HA2LPNLTF. Those 60 HA2LPNLTF were subsequently resume-trained (optimized) for making a better HADWSTE.

The goal of making a set of HA2LPNLTF into HADWSTE for tracking MWS at bad SDIS

As tracking MWS at bad SDIS is possible only under boosting HADWSTE of a set of HA2LPNLTF, a method of DWST resume-training shall be stated. It implies performance optimization of every HA2LPNLTF. After the optimization, HADWSTE shall perform closely to its best.

Selection of HA2LPNLTF out of a set of ordinary GND-trained 2LPNLTF

Suppose there is an aggregate of ordinary GND-trained 2LPNLTF, any of which cannot cope singly with tracking MWS at bad SDIS. Denote by $\tilde{\rho}(\alpha)$ the averaged tracking error rate (TER) of the α -th 2LPNLTF along with its TER $\rho(\sigma_H;\alpha)$ at SDIS maximum. Selection of HA2LPNLTF out of the aggregate lies in accumulating such 2LPNLTF whose performance (in percentage) is either $\tilde{\rho}(\alpha) \le \tilde{\rho}_{\max}$ or $\rho(\sigma_H;\alpha) \le \rho_{\max}^{\langle H\rangle}$ for some defined $\tilde{\rho}_{\max}$ or $\rho_{\max}^{\langle H\rangle}$ (Fig. 1). The selected 2LPNLTF is resume-trained until its performance is bettered enough.

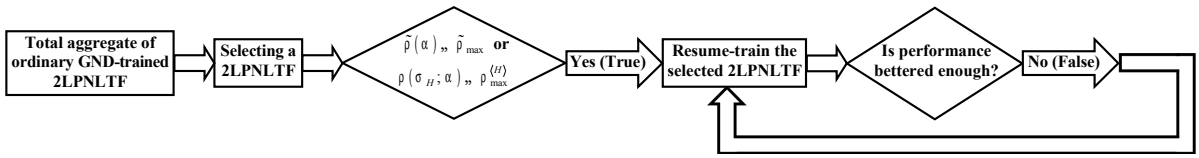

Fig. 1 – Selection of an HA2LPNLTF out of the aggregate of ordinary GND-trained 2LPNLTF, and resume-training it subsequently

Note that it is hard to know whether the performance could be bettered further. Therefore, the number of cycles of the resume-training will be limited. It is adjusted for each problem of tracking MWS at bad SDIS specifically. The same concerns those limits $\tilde{\rho}_{\max}$ and $\rho_{\max}^{\langle H\rangle}$ to sort out 2LPNLTF into HA2LPNLTF and others.

Cyclic resume-training of the selected HA2LPNLTF

Before boosting B HA2LPNLTF, they are resume-trained cyclically under the following parameters. $c_{\text{outer}} \in \mathbb{N} \setminus \{1\}$ is number of outer cycles of repeating the resume-training over all B HA2LPNLTF, and an HA2LPNLTF is resume-trained for $c_{\text{inner}} \in \mathbb{N} \setminus \{1\}$ inner cycles. The resume-training implies passing additional training sets $Y_R^{\langle H\rangle}$ in (1) through HA2LPNLTF for $A \in \mathbb{N}$ times. While passing, if the HA2LPNLTF performance is not bettered for $D \in \mathbb{N} \cup \{0\}$ times by $D < A$ then the inner cycle is broken, HA2LPNLTF is not updated, and the next inner cycle begins. It runs while $\tilde{\rho}^*(\alpha_*) \ldots \tilde{\rho}(\alpha_*)$ by the current TER $\tilde{\rho}^*(\alpha_*)$ of the α_*-th HA2LPNLTF, but no longer than for c_{inner} cycles. An outer cycle is ended with saving the set of the optimized HA2LPNLTF. The saved set $C_{\text{HA2LPNLTF}}$ is re-run within the inner cycles for c_{outer} times (Fig. 2).

```
sigma_H = 0.12;   mu = 1.5;   % -------------------------  Parameters of SDIS
R = 1;   H = 10;              % ------------  Parameters of the training set
test = 400;                   % ------------  Size of the testing sample set
tracker_trainParam = [1 750*3 Inf 0.6 1e-06 6 0.01 1.05 0.7 1.04];
load C_HA2LPNLTF     % -------------------------  Loading a set of HA2LPNLTF
B = size(HA2LPNLTF, 2);   c_outer = 5;   c_inner = 12;   A = 10;   D = 4;
figure (1),   plot(1:B, mean(TER, 2), 'r.-'),   hold on
for outer_cycle_counter = 1:c_outer
    load C_HA2LPNLTF     % -------------------  Re-loading a set of HA2LPNLTF
    TER_current = TER;   % ---------------  Take TER of the current HA2LPNLTF
    HA2LPNLTF_current = HA2LPNLTF;   % ----------  Take the current HA2LPNLTF
    for a_star = 1:B
        tracker_pretrained = HA2LPNLTF(a_star);  % -------  Take an HA2LPNLTF
        inner_cycle_counter = 0; % ---  Start an inner cycle for an HA2LPNLTF
        while mean(TER_current(a_star, :)) >= mean(TER(a_star, :))
            inner_cycle_counter = inner_cycle_counter + 1;
            if inner_cycle_counter > c_inner
                break
            end
            tracker_TER_old = TER_current(a_star, :);  % ---  TER for compare
            Classifiers_old = HA2LPNLTF_current(a_star);
            nonbettered_perf_counter = 0;
            for addQpass = 1:A
                [tracker, tracker_TER, TER_varianceMax, ...
                 traintime, trainparam_stop, trainparam_num_epochs, ...
                 t_0retrainparam_stop, t_0retrainparam_num_epochs] = ...
                    restr_HADWSTE(1, tracker_pretrained, tracker_trainParam, ...
                    'traingda_2LP', sigma_H, mu, ...
                    R, H, addQpass, test, 0, 0);   % ----------  Resume-training
                if mean(tracker_TER) >= mean(tracker_TER_old)
                    nonbettered_perf_counter = nonbettered_perf_counter + 1;
                    if nonbettered_perf_counter == D
                        break   % --------  Cycle is broken, HA2LPNLTF is not
                    end   % -------  updated, and the next inner cycle begins
                else
                    tracker_TER_old = tracker_TER;  % ------  TER for compare
                    Classifiers_old = {tracker};
                end
            end
            TER_current(a_star, :) = tracker_TER_old;  % --------  Update TER
            HA2LPNLTF_current(a_star) = Classifiers_old;
        end
        HA2LPNLTF(a_star) = HA2LPNLTF_current(a_star); % --  Update HA2LPNLTF
        TER(a_star, :) = TER_current(a_star, :);       % ----  Update its TER
    end
    save C_HA2LPNLTF HA2LPNLTF TER    % -----  Set of the optimized HA2LPNLTF
    if outer_cycle_counter == c_outer
        plot(1:B, mean(TER, 2), 'g.-')
    else
        plot(1:B, mean(TER, 2), 'k.-')
    end
end
```

Fig. 2 – MATLAB code of a script for cyclic resume-training

of the selected HA2LPNLTF within the set $C_{\text{HA2LPNLTF}}$

The code in Fig. 2 is adjusted for a problem of tracking at $\sigma_0 = 0{,}12$ and $\mu = 1{,}5$ by $R = 1$ and $H = 10$. These magnitudes nonetheless can be changed easily in lines 1 and 2. The parameters of cyclic resume-training c_{inner}, c_{outer}, A, D are changed in line 6 as well. The exampled problem is of tracking 20 MWS by 10 WIF. Note that generalized regression neural network (GRNN) as a kind of radial basis network (RBF, often used for function approximation) solves it at poor TER (which, on average, is 3,92 % at least). Probabilistic neural network (PNN) as a kind of RBF suitable for classification problems, tracks unstably and poorer. RBF itself, in different configurations, doesn't track 20 MWS by 10 WIF at $\sigma_0 = 0{,}12$ and $\mu = 1{,}5$ appropriately.

For the exampled problem, a 2LPNLTF tracker performs at $\tilde{\rho}(\alpha) \cong (2{,}9;\ 4{,}1)$ by 60 neurons in hidden layer and passing the training set $Y_1^{\langle 10\rangle}$ in (1) through 2LPNLTF for 16 times. This is near-optimal 2LPNLTF configuration for that problem. Setting $\tilde{\rho}_{\max} = 3$ over 2000 ordinary GND-trained 2LPNLTF, we got 97 HA2LPNLTF performing at $\tilde{\rho}(\alpha_*) \,„\, 3 \ \forall\ \alpha_* = \overline{1, 97}$. Figure 3 shows results of running the code in Fig. 2.

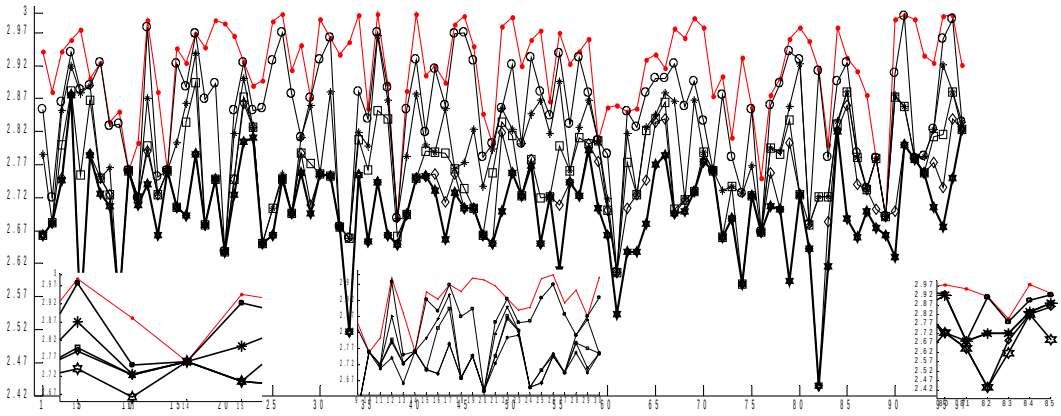

Fig. 3 – Six polylines whose vertices are the decreasing averaged TER, starting from the initial set $C_{\text{HA2LPNLTF}}$ (dots) and descending down through five successively optimized sets of HA2LPNLTF (circles, stars, squares, diamonds, and the best is thicker hexagrams)

For severer selection with TER at 2,9 %, TER decreases tighter (Fig. 4). Predictably, at both Fig. 3 and 4 some points remain unmoved. Altogether there are three such points, where the single unmoved point occurred for severer selection.

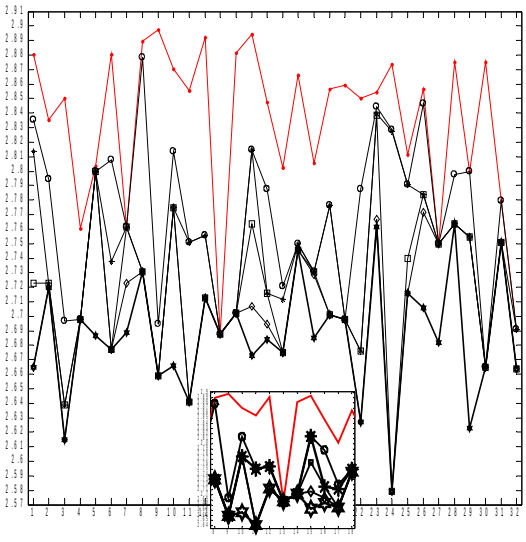

Fig. 4 – Descending TER of 32 HA2LPNLTF for severer selection

Discussion and conclusive remarks

The resume-training method optimizes HA2LPNLTF performance, adjusting heuristically the number of passes of the training set in (1) through HA2LPNLTF. After the optimization, HADWSTE performs closely to its best, even under equally-weighted compositions [2]. Thus the inner cycles' limit should be extended if selection becomes severer. For a problem of tracking 20 MWS by 10 WIF, 97 HA2LPNLTF are selected, whose averaged TER is decreased off 2,92 % down to 2,7 % owing to five cycles of the resume-training. TER of 32 HA2LPNLTF selected by severer conditions, is decreased off 2,83 % down to 2,69 %.

For tracking MWS at bad SDIS, boosting by HADWSTE is far beyond better than RBF, GRNN, PNN, or their combinations. But boosting has its own limit [1, 2]. This limit may be revealed nearly by sufficiently

great numbers of outer and inner cycles, and number of passes of the training set in (1) through HA2LPNLTF. Like boosting, resume-training has its own limit also. Nonetheless the limit, the severer selection of HA2LPNLTF isn't necessarily followed by greater number of TER points remaining unmoved. Here decrement of the averaged TER just grows down.

References

1. Romanuke V.V. Optimizing parameters of the two-layer perceptrons' boosting ensemble training for accuracy improvement in wear state discontinuous tracking model regarding statistical data inaccuracies and shifts / V.V. Romanuke // Problems of tribology. – 2015. – N. 1. – P. 65 – 68.

2. Romanuke V.V. Equally-weighted compositions of Gaussian-noised-data-trained two-layer perceptrons in boosting ensembles for high-accurate discontinuous tracking of wear states regarding statistical data inaccuracies and shifts / V.V. Romanuke // Problems of tribology. – 2015. – N. 2. – P. 53 - 56.

Поступила в редакцію 02.07.2015

Романюк В. В. **Метод донавчання відслідковувачів дискретного стану зносу для складання бустингових високоточних комітетів, необхідних для урахування похибок і зсувів у статистичних даних.**

Для відслідковування станів зносу металу за значних похибок і зсувів у статистичних даних пропонується метод донавчання відслідковувачів дискретного стану зносу з метою їх підсилення у високоточних комітетах. Цими відслідковувачами є двошарові персептрони, навчені на даних з гаусовими шумами. Відбирається звичайний відслідковувач і, якщо його продуктивність задовільна, він циклічно донавчається. Кількість додаткових подач навчальних множин обмежується. Повна процедура донавчання може бути зациклена.

Ключові слова: знос металу, відслідковувач стану зносу, похибки і зсуви у статистичних даних, високоточний комітет бустингу.