OPTIMIZED RECURRENT NEURAL NETWORK-BASED TOOL WEAR MODELING IN HARD TURNING

Xiaoyu Wang and Yong Huang
Department of Mechanical Engineering
Clemson University
Clemson, SC

KEYWORDS
Tool Wear, Hard Turning, Recurrent Neural Network, Extended Kalman Filter

ABSTRACT
Hard turning has been proved to be more effective and efficient than traditional grinding operations in machining hardened steels. However, rapid tool wear is still one of the major hurdles affecting its wide implementation in industry. Better modeling of the tool wear progression helps optimize cutting conditions and/or tool geometry to reduce tool wear, which can make hard turning a feasible technology. The objective of this study is to design a recurrent neural network (RNN)-based estimator for tool wear modeling in hard turning. The extended Kalman filter (EKF) algorithm is applied to train the network. Its structure is further optimized using a pruning algorithm. The proposed estimator is tested in modeling the cubic boron nitride (CBN) tool wear progression in hard turning. Due to its optimized recurrent network connections, the estimator can better model nonstationary and dynamical systems, such as the tool wear progression, in terms of the training speed and generalization ability.

INTRODUCTION
The hard turning process is defined as the single point turning of materials with hardness higher than 50 HRc under the small feed and fine depth of cut condition. It offers possible benefits over the process of grinding in the context of lower equipment costs, shorter setup time, fewer process steps, greater part geometry flexibility, and the elimination of cutting fluid use (Konig et al. 1984; Tönshoff et al. 2000). Among the available tool materials, cubic boron nitride (CBN), second to diamond in hardness and inert to steel materials, has been recommended as the best tool material and widely used for hard turning operations. However, one of the major hurdles preventing the wide implementation of hard turning in industry is the severe wear of CBN tools. The cost of hard turning tools and the tool change downtime due to rapid tool wear can impact the economic viability of precision hard turning. For a given tool and workpiece combination, the ability to estimate the tool wear as a function of cutting conditions - cutting speed, feed rate, and depth of cut, is critical to the overall optimization of a hard turning process.

Tool wear modeling as a general research topic has been researched by numerous
researchers and its overall progress can be mainly classified into four categories as reviewed in Wang et al. (2008a): analytical model-based approach which models tool wear as a function of cutting conditions, cutting environment, tool geometry and property, and/or workpiece property; computational method-based approach which applies the finite element method (FEM) to model tool wear development process; artificial intelligence (AI)-based approach which includes artificial neural networks (ANNs), fuzzy logic, and support vector machine; and parametric model-based approach such as the Taylor tool life model and the regression model. With respect to CBN tool wear modeling in hard turning, the main endeavors include analytical model-based approach (Huang and Liang 2004a; Huang and Liang 2004b); AI-based approach (Ozel and Karpat 2005); and the Taylor tool life equation-based approach (Poulachon 2001; Dawson 2002).

Although analytical models can provide better insight to the underlying physical wear progression mechanisms, they sometimes are less accurate because they often apply simplifications and assumptions in their modeling development process (Scheffer et al. 2003). The FEM approach also provides insight to the process; however, it is too time consuming and not suitable for optimization using the current computing technology. Time series (Boyd et al. 1996) and regression models (Ozel and Karpat 2005) are typically less accurate when compared with ANNs. If both accuracy and speed are of interest instead of the underlying wear progression mechanisms, AI-based modeling approaches are favored for real applications. Among the AI-based approaches, ANN is a viable, reliable, and attractive approach for tool wear modeling (Chryssoulouiris and Guillot 1990; Haber and Alique 2003) due to the following reasons: 1) ANNs have strong strength in modeling nonlinear process which makes them suitable for modeling the tool wear process; 2) The data driven feature of ANNs makes them powerful in parallel computing and capable of handling large amount of data; and 3) ANNs are good at fault tolerance and adaptability and suitable for modeling tool wear during metal cutting process which always subjected to noisy environment.

Among the ANNs approaches, the multilayer perceptron NN (MLP) is the most frequently used one. Previous research has found that a fully forward connected NN (FFCNN) exhibits better performance in terms of the generalization ability, training speed, and structural robustness than that of an MLP (Wang et al. 2008a). The FFCNN can be further modified into a fully connected recurrent neural network (RNN), accommodating internal or external recurrent connections among neurons. Unlike an FFCNN, a RNN can store information from past states, making it more capable of modeling nonstationary and dynamical phenomena. The RNN excels the FFCNN in faster training speed and better generalization ability in modeling nonstationary and dynamical systems.

The goal of this study is to develop a RNN-based estimator with application to CBN tool flank wear modeling. The extended Kalman filter (EKF) algorithm is utilized to train the network. Network connectivity optimization is achieved by disconnecting some less important connections based on a connectivity pruning algorithm. The modeling performance of RNN and optimized RNN is compared with that of an MLP, an FFCNN, and an optimized FFCNN. The RNN approach has proved to be faster, more accurate, and more robust. The study will contribute to better optimize the cutting conditions and tool geometry for hard turning as well as other metal cutting processes.

The paper first introduces the theoretical background of the proposed RNN modeling approach and then compares the performance of different ANN approaches including RNN in CBN hard turning experimental studies (Huang and Liang 2004a). Finally, some conclusions are made regarding the proposed approach.

BACKGROUND

CBN Tool Wear in CBN Hard Turning

The cutting edge of a tool insert in machining is subject to a combination of high stresses, high temperatures, and perhaps chemical reactions which would result in the development of tool wear. These mechanisms depend on the tool and workpiece material combination, cutting geometry, environment, and mechanical and thermal loadings encountered. Different classifications of tool wear processes have been addressed in the literature. Basically, five wear mechanisms or any combinations of them are involved in the tool wear progression. They are
abrasion, adhesion, fatigue, dissolution/diffusion, and tribochemical process. It is well accepted that the tool wear mechanisms in machining involve more than one wear mechanism and it is difficult to predict the relative importance of any one of them (Huang, Chou, and Liang 2007). Crater and flank wear are the most reported wear patterns in machining including hard turning. Crater wear is mainly caused by physical, chemical, and/or thermomechanical interactions between the rake face of the insert and the hot metal chip, and flank wear occurs primarily when the flank face rubs against the workpiece surface.

CBN tool flank wear length or wearland (VB), as shown in Figure 1 which is drawn based on a typical CBN tool wear observation (Dawson 2002), is generally regarded as the tool life criterion or an important index to evaluate the tool performance in hard turning. The tool wear rate is assumed uniform across the width of cut. The main wear mechanisms in CBN turning hardened steels are generally considered to be a combination of abrasion, adhesion, and diffusion; and the contribution of each wear mechanism is related to cutting conditions, tool geometry, and material properties of the tool and the workpiece (Huang and Liang 2004a).

**Recurrent Neural Network**

The recurrent neural network is proposed in this study to model the tool wear progression in hard turning. Its architecture, optimization algorithm, and training algorithm are introduced in the following sections.

**Recurrent Neural Network Architecture.** An FFCNN proposed by Werbos (1990) is adopted as the backbone of the proposed RNN. As shown in Figure 2, the RNN is comprised of m neurons in its input section, h neurons in its hidden section and n neurons in its output section. The proposed RNN is modified from an FFCNN by accommodating the intra-neuron internal recurrency (the dashed lines in Figure 2) in its hidden section. In addition to the forward connections in FFCNN, each neuron in its hidden section also takes feedback connections from the neurons right to it. Hence, the RNN is fundamentally different from the feedforward NN in the sense that it not only operates on the input space but also on the internal state space.

**Recurrent Neural Network Architecture.**

For each neuron $i$, its output at the time step $k$, $y_i^*(k)$, is determined by the neuron’s activation function $f_i(\cdot)$ and net input $net_i(k)$:

$$y_i^*(k) = f_i(net_i(k)), \quad 1 \leq i \leq m + h + n$$

(1)

The net input for neuron $i$ in the input and output sections is defined as:

$$net_i(k) = \sum_{j=1}^{m-1} w_{ij} y_j^*(k) \quad i \notin \{m+1, m+h\}$$

(2)

where $w_{ij}$ is the element of the weight matrix $W$ accounting for the connection weight from neuron $j$ to neuron $i$. Due to the feedback connections introduced in the hidden neuron section, the net input of each neuron in this section is comprised of two parts, the summation of outputs at the current step from the neurons left to it and the summation of outputs at the previous step of hidden neurons right to it for this one-step recurrence network.

$$net_i(k) = \sum_{j=1}^{m} w_{ij} y_j^*(k) + \sum_{j=m+1}^{m+h} w_{ij} y_j^*(k-1) \quad m < i \leq m + h$$

(3)

where $n = m + h + n$ is the total number of neurons of the network, and $y_j^*$ is the output of $j$th neuron.
process is illustrated in Figure 3, where the RNN originally has a structure of 1-3-1, and two connections (c31 and c24) are disconnected after optimization.

**FIGURE 3. AN ILLUSTRATION OF CONNECTIVITY OPTIMIZATION.**

**EKF Training Algorithm.** The EKF algorithm was first introduced to train neural networks by Singhal and Wu (1989). Compared to the most widely used back-propagation (BP) algorithm, the EKF learning algorithm has the following advantages: 1) the EKF algorithm helps to reach the training steady state much faster for nonstationary processes (Zhang 2005); and 2) the EKF algorithm excels the BP algorithm when the training data is limited (Puskorius and Feldkamp 1994). Hence the EKF learning algorithm (Puskorius and Feldkamp 1994; Haykin 1999) is favored here, and the EKF-based network weight updating at time step k is introduced as follows.  

\[
\hat{w}(k) = \hat{w}(k-1) - K(k) \left( y(k) - \hat{y}(k) \right) \tag{4}
\]

\[
K(k) = P(k-1)H(k) \left[ R(k) + H(k)^T P(k-1) H(k) \right]^{-1} \tag{5}
\]

\[
P(k) = P(k-1) - K(k)H(k)^T P(k-1) + Q(k) \tag{6}
\]

where \( \hat{w} \) is the weight estimate, which is formed from the weight matrix (Wang et al. 2008a), \( \hat{y} \) is the network output vector, \( \hat{y} \) is the measurement vector, \( K \) is the Kalman gain matrix, \( H \) is the derivative matrix of the network outputs with respect to the trainable network weights, \( R \) is the covariance matrix of measurement noise, \( Q \) is the covariance matrix of process noise, \( P \) is the covariance matrix of the estimation error (Puskorius and Feldkamp 1994), and \( k-1 \) means one time step before time step \( k \).

The RNN training process is depicted in Figure 4. First the parameters of EKF, \( P \), \( Q \), and \( \hat{w} \) are initialized. Then the training data are fed into the network, and the orderly derivatives of the network’s outputs with respect to the weight vectors \( \left( \frac{\partial \hat{y}}{\partial \hat{w}} \right) \) are calculated using the chain rule (Wang et al. 2008a), forming a Jacobian matrix \( H \). The derivatives take different forms depending on the specific weight connections. Finally, the Kalman filter equations (4)–(6) are applied to train the network weights until the specified stop criteria are met.

**FIGURE 4. RNN TRAINING PROCEDURE.**

**EXPERIMENTAL VALIDATION**

**Experimental Setup**

To better appreciate the validity in applying the proposed RNN estimators for CBN tool wear modeling, hardened AISI 52100 bearing steel with a hardness 62 HRc was machined on a horizontal Hardinge lathe using a low CBN content tool insert (Kennametal KD050) with a -20º and 0.1 mm wide edge chamfer and a 0.8 mm nose radius. The ISO DCLNR-164D tool holder was used, which introduced a negative 5º rake angle. No cutting fluid was applied. Flank wear length was measured using an optical microscope (Zygo NewView 200). The experiment was stopped when a sudden force jump was observed signaling a chipping or broken tool condition.

**TABLE 1. CUTTING CONDITIONS OF EXPERIMENTS.**

<table>
<thead>
<tr>
<th>Condition index</th>
<th>Speed (m/s)</th>
<th>Feed (mm/re)</th>
<th>Depth of cut (mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3.05</td>
<td>0.152</td>
<td>0.203</td>
</tr>
<tr>
<td>2</td>
<td>1.52</td>
<td>0.152</td>
<td>0.203</td>
</tr>
<tr>
<td>3</td>
<td>3.05</td>
<td>0.076</td>
<td>0.203</td>
</tr>
<tr>
<td>4</td>
<td>2.29</td>
<td>0.114</td>
<td>0.203</td>
</tr>
<tr>
<td>5</td>
<td>1.52</td>
<td>0.076</td>
<td>0.203</td>
</tr>
<tr>
<td>6</td>
<td>3.36</td>
<td>0.114</td>
<td>0.203</td>
</tr>
<tr>
<td>7</td>
<td>2.29</td>
<td>0.061</td>
<td>0.203</td>
</tr>
<tr>
<td>8</td>
<td>2.29</td>
<td>0.168</td>
<td>0.203</td>
</tr>
<tr>
<td>9</td>
<td>1.21</td>
<td>0.114</td>
<td>0.203</td>
</tr>
<tr>
<td>10</td>
<td>2.29</td>
<td>0.114</td>
<td>0.203</td>
</tr>
<tr>
<td>11</td>
<td>1.52</td>
<td>0.076</td>
<td>0.102</td>
</tr>
<tr>
<td>a</td>
<td>1.52</td>
<td>0.076</td>
<td>0.152</td>
</tr>
<tr>
<td>b</td>
<td>1.52</td>
<td>0.076</td>
<td>0.152</td>
</tr>
</tbody>
</table>

Transactions of NAMRI/SME  Volume 37, 2009
Machining test was performed based on a standard central composite design test matrix with an alpha value of 1.414. The center point (0,0) was determined based on the tool manufacturer’s recommendation (Huang and Liang 2004a). A typical depth of cut was suggested as 0.203 mm, which was used in the test matrix. To further investigate the effect of depth of cut on tool wear, experiments with various depths of cut were also studied. Ten different cutting conditions (Huang and Liang 2004a), namely conditions 1-5, 8-10, a, and b are listed in Table 1. Conditions 7 and 11 are not utilized here since they are the same as condition 4, and condition 6 (cutting speed = 3.36 m/s) is also not used since the break-in period accounted for a large portion of tool flank wear and microchipping was a dominant factor of tool life under such an aggressive cutting speed. Uncertainty characterization is not offered here due to the size of the experimental data set.

For a given tool and workpiece combination, the capability to estimate the tool wear as a function of cutting conditions is critical to the overall optimization of a hard turning process. The CBN tool wear progression in hard turning is modeled using the RNN as follows.

RNN-Based Estimator Implementation

In this study, the RNN and the optimized RNN (OptRNN) are formed to model the CBN tool wear progression in hard turning and their modeling performance is compared with the measurements as well as that of the fully forward NN (FFNN) approaches in a previous study (Wang et al. 2008a).

Training and Testing Data Preparation. As shown in Table 1, there are total 10 groups of data available from the hard turning experiment. Among them data of conditions 1, 5, 9, 10, and a are used for network training. Each tool wear measurement pair (time versus tool wear) is treated as a training pattern, resulting in 48 training patterns under conditions 1, 5, 9, 10, and a (five conditions in total). The rest data are used to test the generalization ability of the proposed RNN estimators.

Training Parameters Configuration. To train the RNN, some training parameters such as P, Q, and R need to be initialized first. The training parameters configuration is referred from a previous study (Puskorius and Feldkamp 1994). The error covariance matrix P is initialized as a diagonal matrix and each of its diagonal elements is initialized as 100. Each diagonal element of the process noise covariance matrix Q is initialized as 0.01 and this value descends linearly within 100,000 training cycles until Q reaches a minimum limit of 0.000001. Similarly, each diagonal element of the measurement noise covariance matrix R is initialized as 100 and it also descends linearly until it reaches a minimum boundary of 2. Both the settings of R and Q help the training error converge to a global minimum.

Training Process Configuration. First, each weight of the network is randomly initialized in the region of [-1, 1]. Training parameters are initialized as mentioned before. Training data are then fed into the EKF training equations (4)–(6) to train the network weights. During the training process, the training data are used for each training epoch and the weights are updated accordingly. The procedure of training using all the training patterns once is called a training step or epoch. The training process stops when the stop criteria are satisfied. The stop criteria are determined by trial-and-error: 1) the number of training step should be less than 500 and the training process stops after 500 steps if no other stop criteria are met; or 2) if the training error is less than 0.03 and the difference between the current error and the error of 20 epochs before is less than 0.0003. A normalized sum of square error (SSE) is used to represent the training error at time step j:

\[ e(j) = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \times 100 \% \]

where T is the matrix transpose operator.

Network Structure. The RNN network structure is first determined by setting the numbers of input neurons, hidden neurons and output neurons. Four independent variables, cutting speed, feed rate, depth of cut and machining time, and a constant bias 1 are used as the inputs. The output is the tool flank wear length. The number of hidden neurons is determined by a trial and error method. According to a rule of thumb (Schalkoff 1997), 12 hidden neurons are chosen first and the training error is found to be 4.2%. Afterward, networks with fewer hidden neurons are selected and the corresponding training errors are investigated. It is found that the RNN with
more than 1 hidden neuron is able to adequately model the tool wear progression. For example, the training error of the network (5-2-1) with 2 hidden neurons (0.045) is quite close to that of the network (5-12-1) with 12 hidden neurons (0.042). However, for the network with 1 hidden neuron, the training error becomes relatively large (0.1282). The simple network structure would reduce the risk of over-fitting, so the network with a 5-2-1 structure as shown in Figure 5 is selected in this study.

![Diagram showing the network structure with 5 input neurons, 2 hidden neurons, and 1 output neuron.](image)

**FIGURE 5. MODELING OF TOOL WEAR PROGRESSION USING THE 5-2-1 RNN.**

### Modeling Performance Comparison

#### Training Results.

Training results indicate the fitness of the network model in modeling the training data. Training data are assumed to be able to represent the overall characteristics of the system being studied. Therefore, from the training process, a network capable of modeling the training data is expected to represent the system dynamics.

During the training process, the appropriate network architecture should be determined first as stated in the previous section. The benchmark MLP has been found to be 5-5-1 and the other networks (RNN and OptRNN) are 5-2-1. The same training data (conditions 1, 4, 5, 9, and a) have been used to train the networks. 500 training steps are used for RNN and OptRNN training while 10000 training steps are used for MLP training since MLP converges much slower.

From the results, all the training errors are smaller than 5% (MLP: 4.8%, RNN: 4.5%, and OptRNN: 4.4%). Figure 6 shows some representative training result comparisons. It can be seen that the modeling performance of the investigated ANNs are close in modeling the training data and all the networks are able to accurately represent the training data and model the tool wear progression.

It should be pointed out that conditions 10 and a are modeled more accurately than conditions 1 and 5. It is because that during this pattern learning process the network is trained orderly from condition 1 to condition a. As a result, more training effort has been put to the most recent training cases while the training error is counted for the overall error for all the training data. While the overall training error of the OptRNN is generally smaller than that of MLP, the OptRNN may have larger errors for some specific training data points.

![Graphs showing training results for conditions 1, 5, 10, and a.](image)

**FIGURE 6. MODELING RESULTS IN TRAINING.**

#### Testing Results.

The trained networks are further tested for their generalization ability. Conditions 2, 3, 4, 8, and b have been used as the testing cases, which are unseen in the network development process.

![Graphs showing testing results for conditions 2, 3, 4, 8, and b.](image)

**FIGURE 7. MODELING RESULTS IN TESTING.**
Figure 7 shows the testing result comparisons among the experimental measurements and the predictions from MLP, RNN, and OptRNN. It can be seen that:

1) Except a few testing data points in condition 4, the RNNs predict much more accurately than MLP for this nonstationary and dynamical tool wear progression;

2) For most cases, the discrepancy between the network prediction and its desired output (the experimental measurement) increases with time;

3) The RNN and OptRNN have the similar modeling performance in this tool wear study mainly due to their inherent recurrent architectures; and

4) The MLP tends to over estimate the tool wear length for all the testing cases which implies its limitation in modeling this nonstationary and dynamical system.

The modeling performance is further compared with that of FFCNN and OptFFCNN (Wang et al. 2008a). Table 2 shows the testing errors for these networks which better illustrate their overall modeling capability. Some observations can be drawn as follows:

1) The average testing errors of optimized networks (Optimized FFCNN and Optimized RNN) are smaller than those of their corresponding networks (FFCNN and RNN);

2) The average testing errors of recurrent networks (RNN and Optimized RNN) are smaller than those of purely forward networks (MLP and FFCNNs); and

3) From the variance of errors, the MLP has the largest variation which means it is the least robust network while the optimized networks have smaller variances (8.0 and 6.1) which indicates the optimization process can improve network’s robustness as well as their modeling accuracy.

TABLE 2. MODELING ERROR COMPARISONS IN TESTING.

<table>
<thead>
<tr>
<th>Condition Index</th>
<th>MLP (%)</th>
<th>FFCNN (%)</th>
<th>Optimized FFCNN (%)</th>
<th>RNN (%)</th>
<th>Optimized RNN (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>8.37</td>
<td>10.65</td>
<td>10.29</td>
<td>10.63</td>
<td>7.65</td>
</tr>
<tr>
<td>3</td>
<td>39.45</td>
<td>26.32</td>
<td>12.77</td>
<td>16.42</td>
<td>10.27</td>
</tr>
<tr>
<td>4</td>
<td>17.94</td>
<td>22.03</td>
<td>11.51</td>
<td>16.09</td>
<td>12.62</td>
</tr>
<tr>
<td>8</td>
<td>15.57</td>
<td>15.27</td>
<td>13.71</td>
<td>9.35</td>
<td>8.65</td>
</tr>
<tr>
<td>B</td>
<td>5.96</td>
<td>6.50</td>
<td>5.60</td>
<td>5.84</td>
<td>5.29</td>
</tr>
<tr>
<td>Average Error</td>
<td>17.46</td>
<td>16.15</td>
<td>10.78</td>
<td>11.67</td>
<td>8.90</td>
</tr>
<tr>
<td>Error Variance</td>
<td>140.4</td>
<td>52.4</td>
<td>8.0</td>
<td>16.5</td>
<td>6.1</td>
</tr>
</tbody>
</table>

DISCUSSION

As stated, training parameter setting in EKF training here has used the parameter setup from a previous study (Puskorius and Feldkamp 1994). These R and Q settings just provide an intuitive guide for the EKF parameter configuration. Improper choice of these parameters may lead to training divergent problems. For example, a too small R will lead to an over large learning rate which causes the training process divergent. A more reliable and systematic approach should be developed to configure these EKF training parameters.

While the proposed approach is data-driven, it can be further developed into as a hybrid modeling approach by seamlessly integrating the RNN powerful nonlinear modeling capacity and the process physical models as in (Wang et al. 2008b), which contains analytical models and neural networks.

CONCLUSIONS AND FUTURE WORK

The RNN-based estimator is proposed to model the CBN tool wear progression in hard turning. The EKF algorithm is applied to train the network and a pruning technology is used to optimize the RNN structure and to form the optimized RNN-based estimator. The modeling performance of the RNN-based estimator has been evaluated using the experimental measurements as well as other ANN architectures and the comparisons show that the RNN-based estimator excels others in training accuracy and generalization ability. Also, it is found that through the connectivity optimization process, the optimized estimator get further robustness improvement in modeling the tool wear progression. Generally speaking, this proposed RNN-based tool wear estimator has been proved to be faster and more accurate when compared with the other network approaches investigated, and it can help better optimize the cutting conditions in hard turning as well as other machining processes.

The future work is to study the training convergence of the proposed RNN estimator. Some challenges may include as follows: 1) how to avoid training divergence, and 2) how to accelerate the convergence by adapting the noise parameters R and Q. How to determine or estimate these two parameters is of great
importance in implementing such RNN-based estimators.

ACKNOWLEDGMENTS

The authors would like to acknowledge the financial support from the South Carolina Space Grant Consortium and the NASA Ames Research Center.

REFERENCES


