Electronic Supplementary Materials

For https://doi.org/10.1631/jzus.A2500549

Adaptive adjustment strategy for EPB operation parameters for pre-control of shield tunneling-induced surface settlement: a case study

Dongsheng WEI, Haibin WEI, Zipeng MA, Heting WEI, Lijie SUN, Xiaokun YU

College of Transportation, Jilin University, Changchun 130000, China

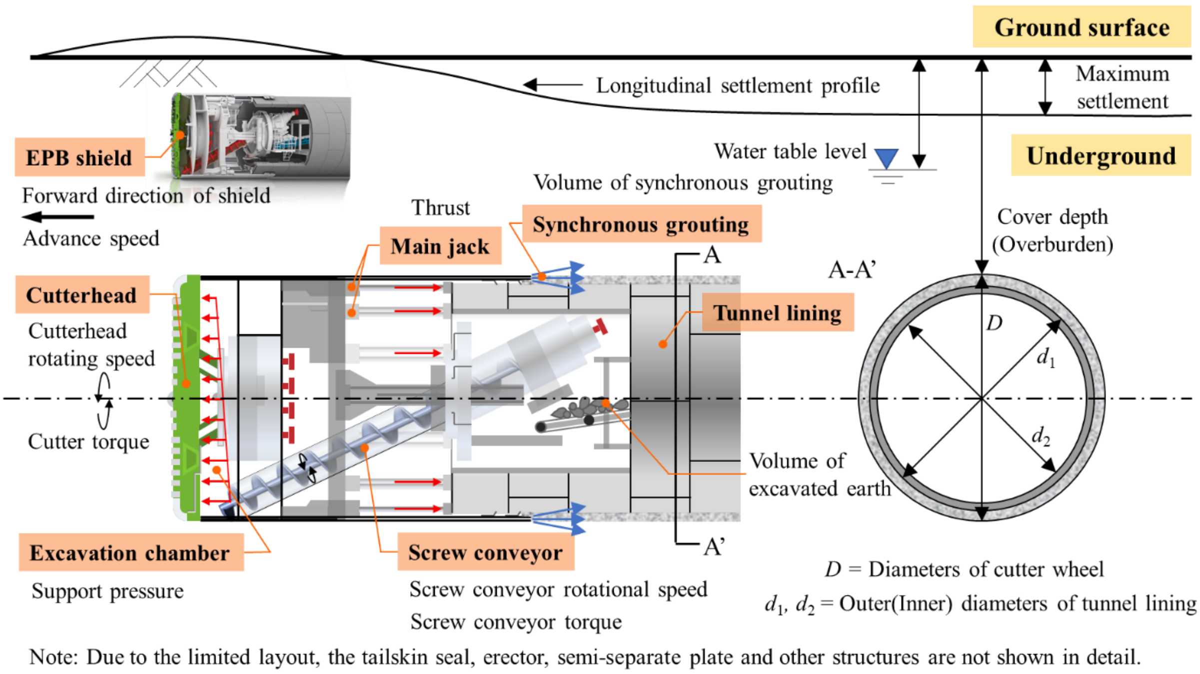


Fig. S1 Schematic view of EPB shield advance

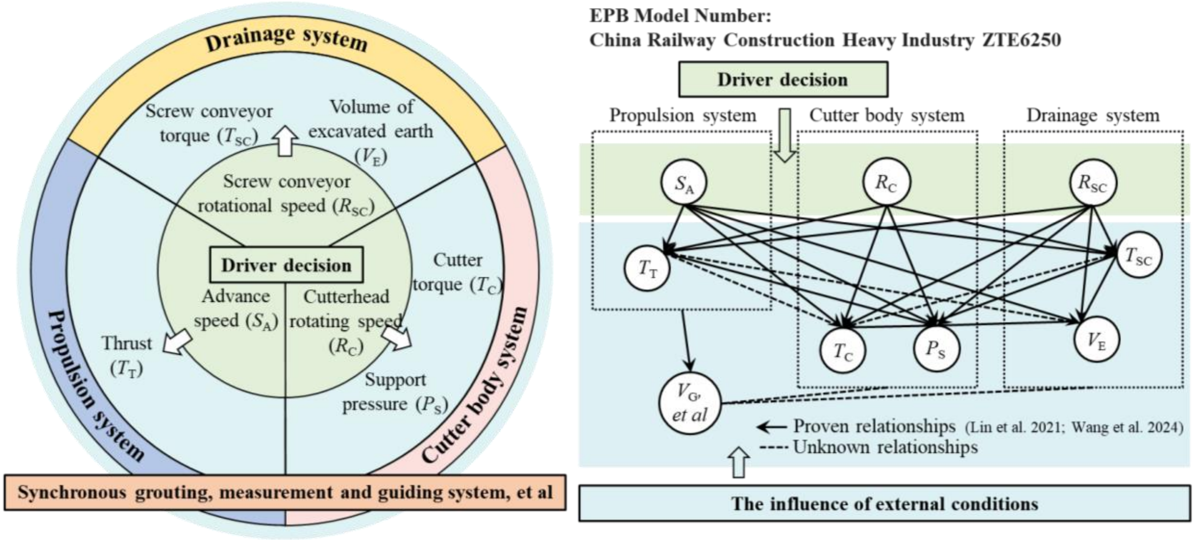
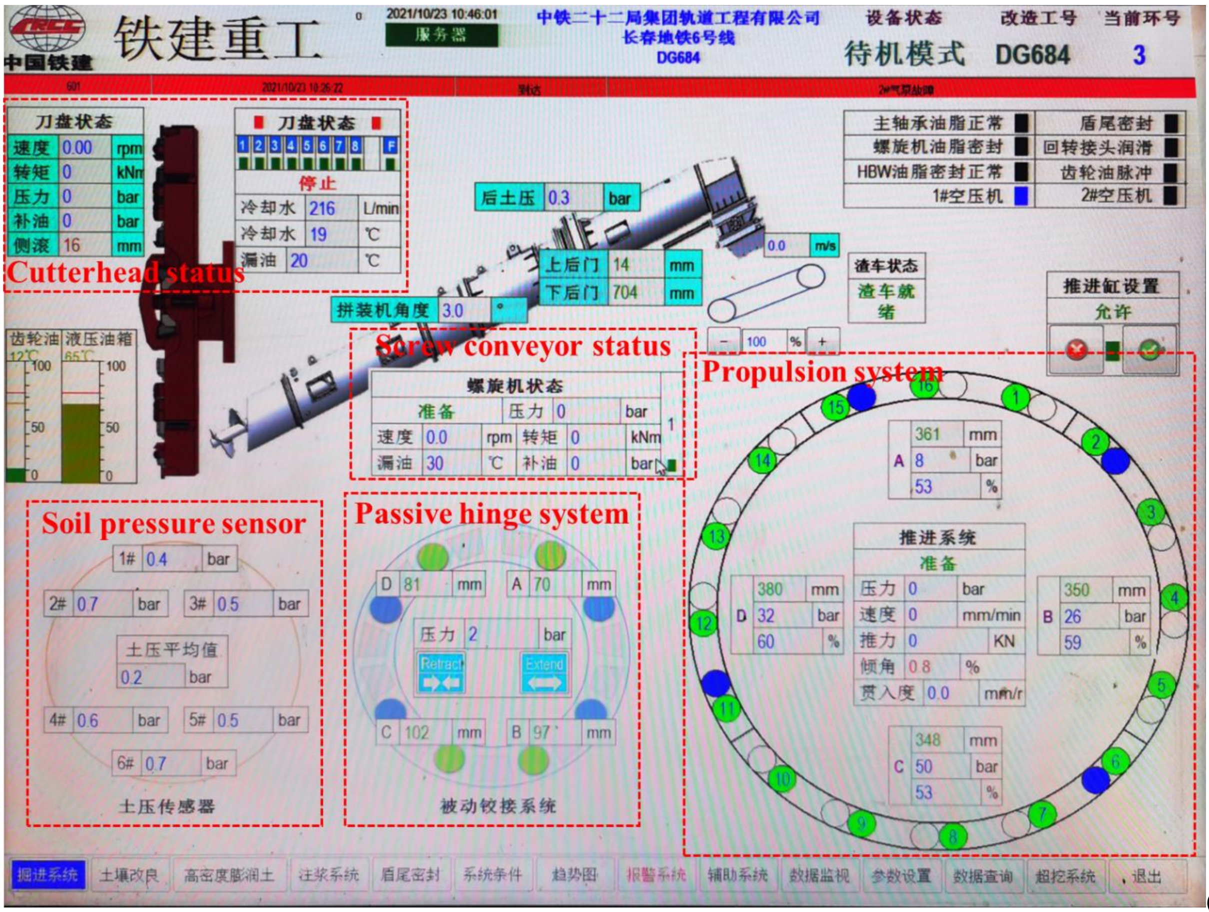
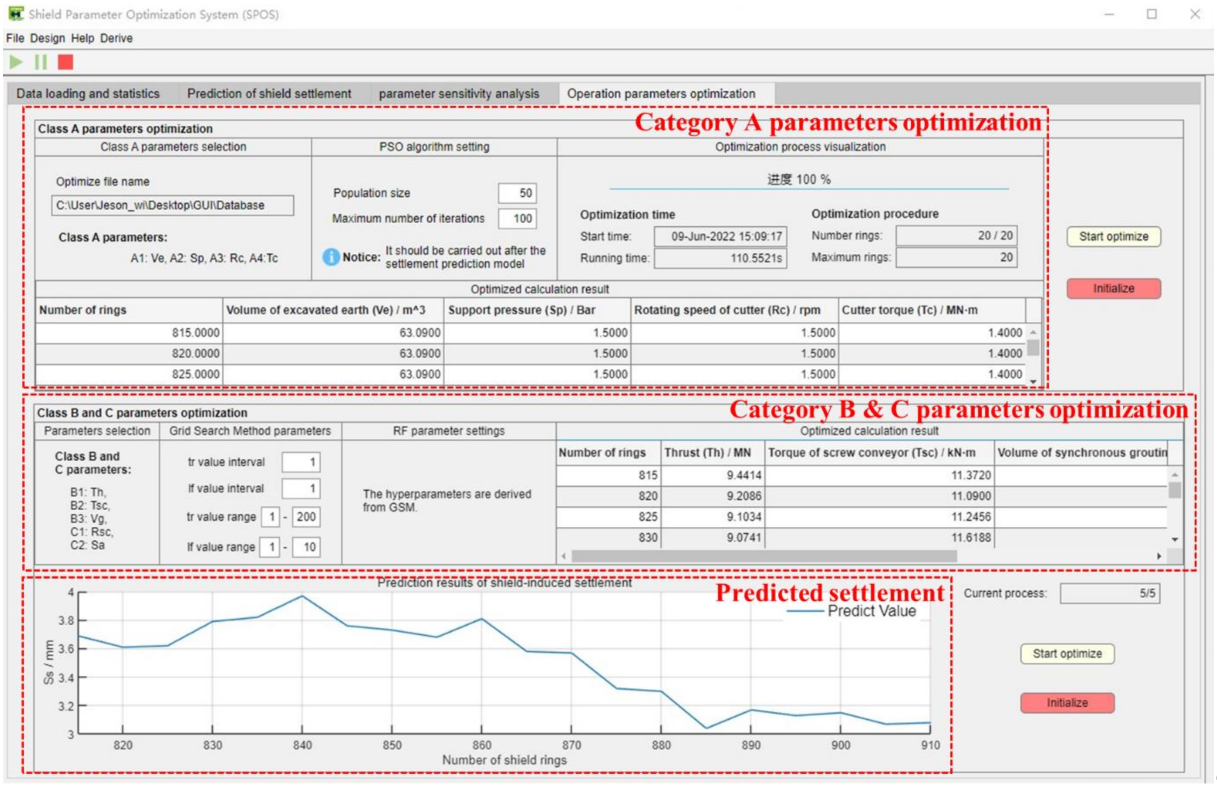


Fig. S2 System relations and interaction topology of parameters in an EPB shield



(a)



(b)

Fig. S3 Interface of (a) EPB shield construction monitoring and (b) SPOS operation

Section S1. Bidirectional long short-term memory neural networks

The LSTM model employs a cell architecture to maintain long-term states, incorporating a 'Gate' memory mechanism. By augmenting the network with control units and internal memory components, the LSTM model is endowed with the functionality to either discard or incorporate information into the cell. The forget gate (f_t) selectively retains the information in the cell state from the previous time step through Eq. (S1), filtering out the relevant content that needs to be passed to the neural network at the current moment. The output of the forget gate is calculated by combining the output of the previous cell and the current input data. The input gate (i_t) determines the amount of new information to be added to the cell state through Eq. (S2), and the tanh layer generates a candidate update vector (Eq. S3). The combination of these two completes the update of the cell state from Eq. (S4), achieving information discard and addition. The output gate (o_t) obtains the output of the current cell state through Eqns. (S5) and (S6). The output is processed by the sigmoid (σ) activation function with relevant weights and input, and then multiplied by the result of the cell state after the tanh activation to obtain the final hidden state. The output of the LSTM model at each time step is given by Eq. (S7). In contrast, the Bi-LSTM model can fully capture the correlations and deep connections between preceding and succeeding data. The output at each time step is calculated using Eq. (S8), as shown in Fig. S4.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{S1}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{S2}$$

$$\tilde{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \tag{S3}$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{S4}$$

$$o_t = \sigma\left(W_o \cdot [h_{t-1}, x_t] + b_o\right) \tag{S5}$$

$$h_t = o_t \odot \tanh\left(C_t\right) \tag{S6}$$

$$O_t = softmax(W_o \cdot h_t + b_o) \tag{S7}$$

$$O_{t-Bi} = softmax(W_o \cdot [\rightarrow h_t, \leftarrow h_t] + b_o) \tag{S8}$$

where, $\tilde{C}_t$ represents the output of the candidate cell; C_t is the previous memory cell; x_t, h_{t-1}, C_{t-1} are input vectors; h_t and C_t are output vectors; b (b_f, b_i, b_c, b_o) are the bias vectors; W (W_f, W_i, W_c, W_o) are weight vectors; σ and tanh are the sigmoid and tanh functions; $\rightarrow h_t$ and $\leftarrow h_t$ represent the hidden states of the forward and backward units at each time step, respectively; $\odot$ denotes element-wise multiplication.

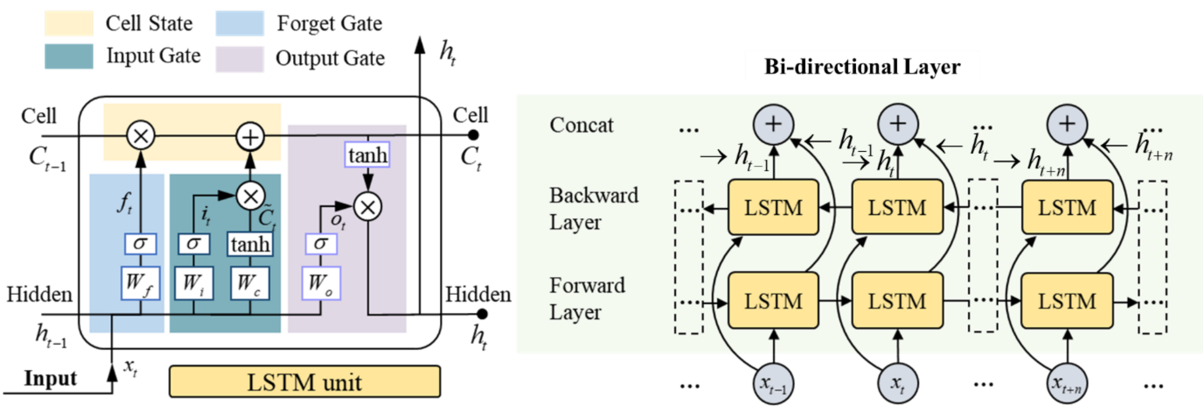


Fig. S4 The architecture of the Bi-LSTM model

Section S2. Multi-head self-attention mechanism

Attention mechanisms, including self-attention (SA) and multi-head self-attention (MHSA), can calculate the autocorrelation and cross-correlation of each element, focusing on elements that make significant contributions to the output. In the self-attention mechanism, the input sequence is first linearly transformed into three matrices: Query (Q), Key (K), and Value (V). Specifically, each element's self-attention score is computed by taking the dot product between its Query vector and all the Key vectors in the sequence. These scores are then normalized using a *softmax* function to generate attention weights, which represent the importance of each element relative to others. The output vector is subsequently derived by weighting the Value vectors with these attention weights through another dot product operation. Finally, the output of the self-attention layer is fed into a fully connected layer to integrate the extracted features. The MHSA mechanism extends this framework by applying multiple parallel self-attention operations, known as "*head*." Each head performs the same self-attention computation on a different linear projection of the input, enabling the model to capture diverse patterns and relationships within the data. The outputs from all the heads are concatenated and linearly transformed again to produce the final output. This architecture allows the MHSA mechanism to attend to different parts of the input sequence from multiple perspectives, enhancing its ability to model complex hierarchical structures. The mathematical formulation of this process is detailed in Eqns. (S9) to (S14), and the architectural principle is illustrated in Fig. S5.

$$Q_i = X \cdot \omega_i^Q \tag{S9}$$

$$K_i = X \cdot \omega_i^K \tag{S10}$$

$$V_i = X \cdot \omega_i^V \tag{S11}$$

$$Attention_i\left(Q_i, K_i, V_i\right) = softmax\left[Q_i K_i^T / \left(d_k\right)^{1/2}\right]V \tag{S12}$$

$$h_i = Attention_i\left(Q_i \omega_i^Q, K_i \omega_i^K, V_i \omega_i^V\right) \tag{S13}$$

$$m\left(Q, K, V\right) = Concat\left(h_1, h_2, h_3, \ldots, h_h\right) \times \omega^0 + b^0 \tag{S14}$$

where, X is the input sequence of length N, which represents as a matrix $X \in \mathrm{R}^{N \times d}$, d indicates the dimensionality of each position in the sequence; d_k denotes the dimension of the k^{th} key; i represents the index of the i^{th} head; ω_i^Q, ω_i^K, and ω_i^V represent the weight matrices for the Q, K, and V variables in the i^{th} head, respectively; m represents the total number of "heads;" h_i represents the data matrix of the i^{th} head; and the "*Concat*" represents the operation of merging the results from multiple "*heads*;" $m\left(Q, K, V\right)$ represents the outputs of the attention.

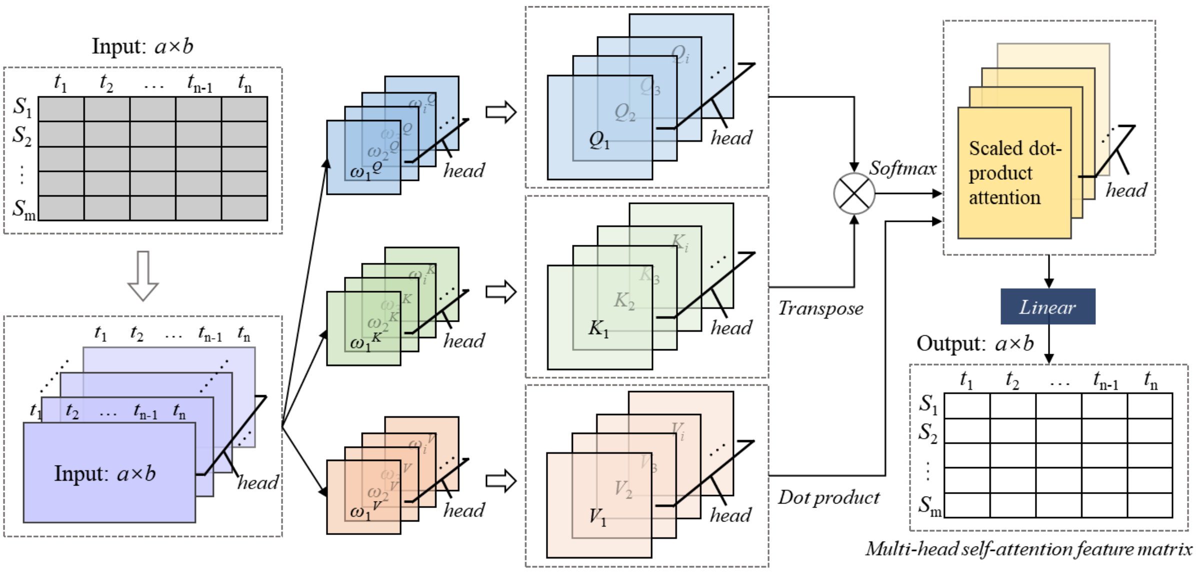


Fig. S5 The principle of the MHSA mechanism

Section S3. Random forest algorithm

The random forest (RF) algorithm is an integrated learning algorithm for tasks such as classification and regression. Two powerful ML techniques, bootstrap aggregation (Breiman, 1996) and random subspace (Tin Kam, 1998) are integrated into the RF algorithm. In bagging, n bootstrap sets are obtained by sampling n training samples and replacing them from the training set. The number of samples and features in the bootstrap is arbitrary and less than the original training set. Each bootstrap set is then constructed as a decision tree. The decision tree classifies the bootstrap set by testing the attributes of the bootstrap set at each node. Each node tests a specific attribute and the leaves of the tree represent the output labels. Moving down a particular branch of the tree tests a particular attribute on each node in order to reach the output label. The final results summarize the outputs of all the leaves.

The output of RF prediction can be expressed as:

$$y = \frac{1}{n_{tree}} \sum_{i=1}^{n_{tree}} y_i(\boldsymbol{x}) \tag{S15}$$

where, y is the average output of a total amount of n_{tree}, and $y_i(\boldsymbol{x})$ is the individual prediction of a tree for an input vector $\boldsymbol{x}$.

Section S4. Particle swarm optimization

The particle swarm optimization (PSO) algorithm, first introduced by Eberhart and Kennedy (Eberhart and Kennedy, 1995), is a probabilistic global optimization algorithm that leverages the cooperation among particles to share information across the entire swarm, thus seeking the optimal solution. PSO operates by iteratively updating the positions and velocity vectors of individual particles, introducing a fitness function to select the extremum values of each particle, and comparing these with the global optimum to determine the optimal group extremum (Hu et al., 2019). The algorithm terminates once an optimal solution is found or after a predefined number of iterations has been reached. The functions for the updated position (χ) and speed (v) of each particle are as follows:

$$v_i^{k+1} = \omega v_i^k + c_1 r_1 (p_{i\text{bset}}^k - \chi_i^k) + c_2 r_2 (g_{\text{best}}^k - \chi_i^k) \tag{S16}$$

$$\chi_i^{k+1} = \chi_i^k + v_i^{k+1} \tag{S17}$$

where, ω signifies the inertia weight, indicative of the capacity to carry forward the velocity from the preceding moment; k represents the iteration count of the particle swarm; $p_{i\text{bset}}$ is the personal best value of an individual particle; g_{best} is the best value observed across the entire swarm; c_1, c_2 are the acceleration coefficients; r_1, r_2 are random variables uniformly distributed between [0,1].

Section S5. Performance indicators

To ascertain the model's predictive capability, Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) are chosen as the metrics for assessing the predictive model, with their calculation formulas detailed subsequently:

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |p_i - r_i| \tag{S18}$$

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (p_i - r_i)^2} \tag{S19}$$

where, n is the count of the data samples; p_i refer to the predicted settlement values; r_i are the measured settlement values. Lower MAE and RMSE values indicate superior model performance.

Table S1 Parameter data set and parameter range for Model 1

Variable (abbreviation)	Parameter type of Model 1	Data			Unit
		Min.	Max.	Ave.	
Advance speed (S_A)	Input	30	65	48.80	mm/min
Cutterhead rotating speed (R_C)	Input	1.5	1.7	1.629	rpm
Cutter torque (T_C)	Input	1.4	6.1	3.399	MN·m
Support pressure (P_S)	Input	1.0	1.5	1.203	bar
Screw conveyor rotational speed (R_{SC})	Input	5.0	12.5	8.049	rpm
Screw conveyor torque (T_{SC})	Input	7	20	12.13	kN·m
Thrust (T_T)	Input	6.5	16	9.182	MN
Volume of synchronous grouting (V_G)	Input	5	7.5	6.573	m^3
Volume of excavated earth (V_E)	Input	63.09	73.33	68.44	m^3
Constrained modulus (E_{oed})	Input	6.80	9.48	8.75	MPa
Cohesion (c)	Input	35.67	50.48	46.89	kPa
Internal friction angle (φ)	Input	21.78	24.66	24.02	°
Lateral earth pressure coefficient at rest (K_0)	Input	0.335	0.389	0.346	-
Cover depth (D_C)	Input	9.45	12.09	10.82	m
Water table level (L_W)	Input	3.25	5.50	4.461	m
Pre-shield arrival surface settlement ($S_{PS7.5}$)	Input	-1.96	1.79	0.112	mm
Pre-shield arrival surface settlement (S_{PS0})	Input	-1.42	3.07	1.084	mm
Shield tunneling-induced surface settlement (S_S)	Output	3.24	8.87	6.147	mm

Negative values for S_{PS0} and $S_{PS7.5}$ indicate ground heave (uplift) caused by face support pressure and grouting pressure transmission ahead of the shield. This is a physically realistic phenomenon in EPB shield tunneling that helps reduce net settlement.

Fig. S6 The detailed view of the cutterhead

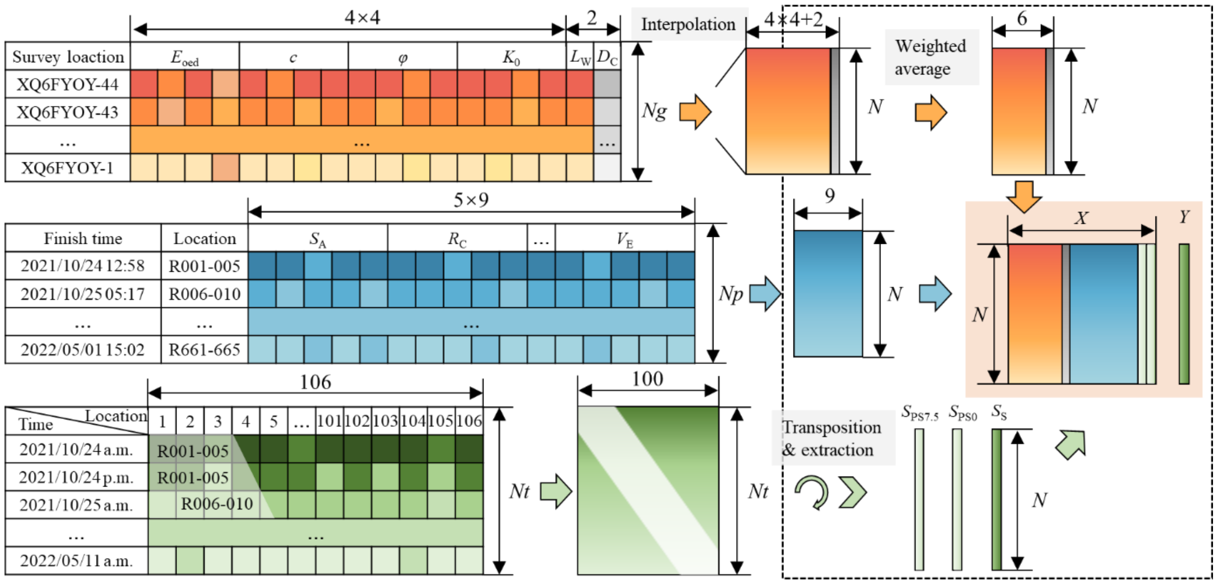


Fig. S7 A representation of the data fusion process

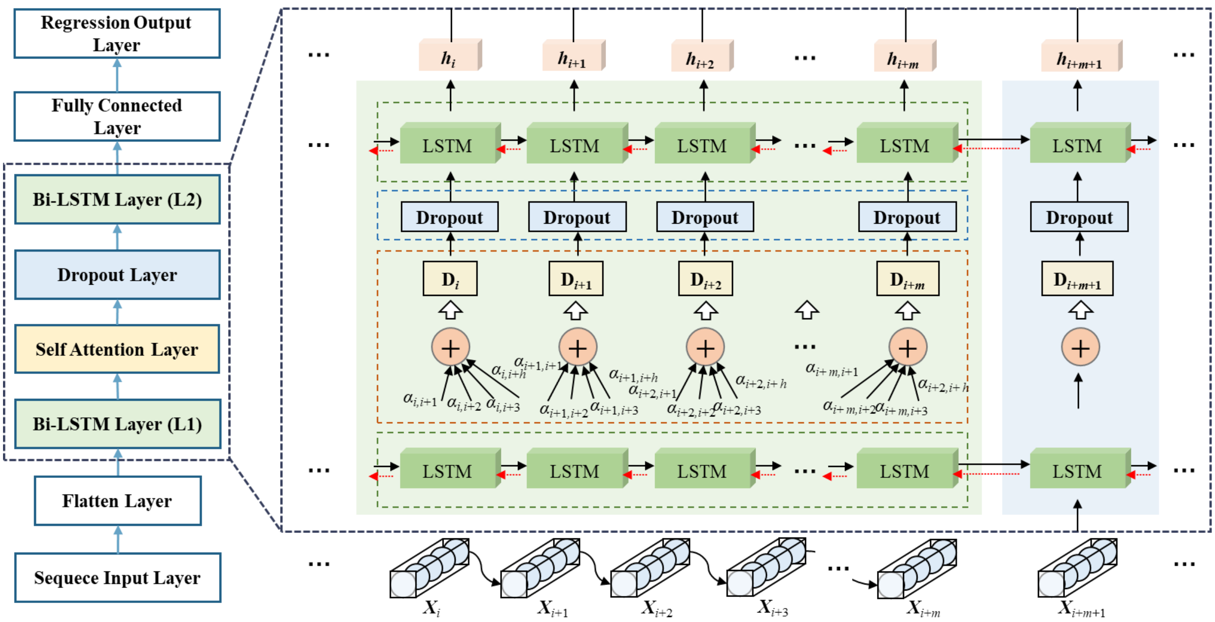


Fig. S8 The architecture of MHSA-Bi-LSTM model

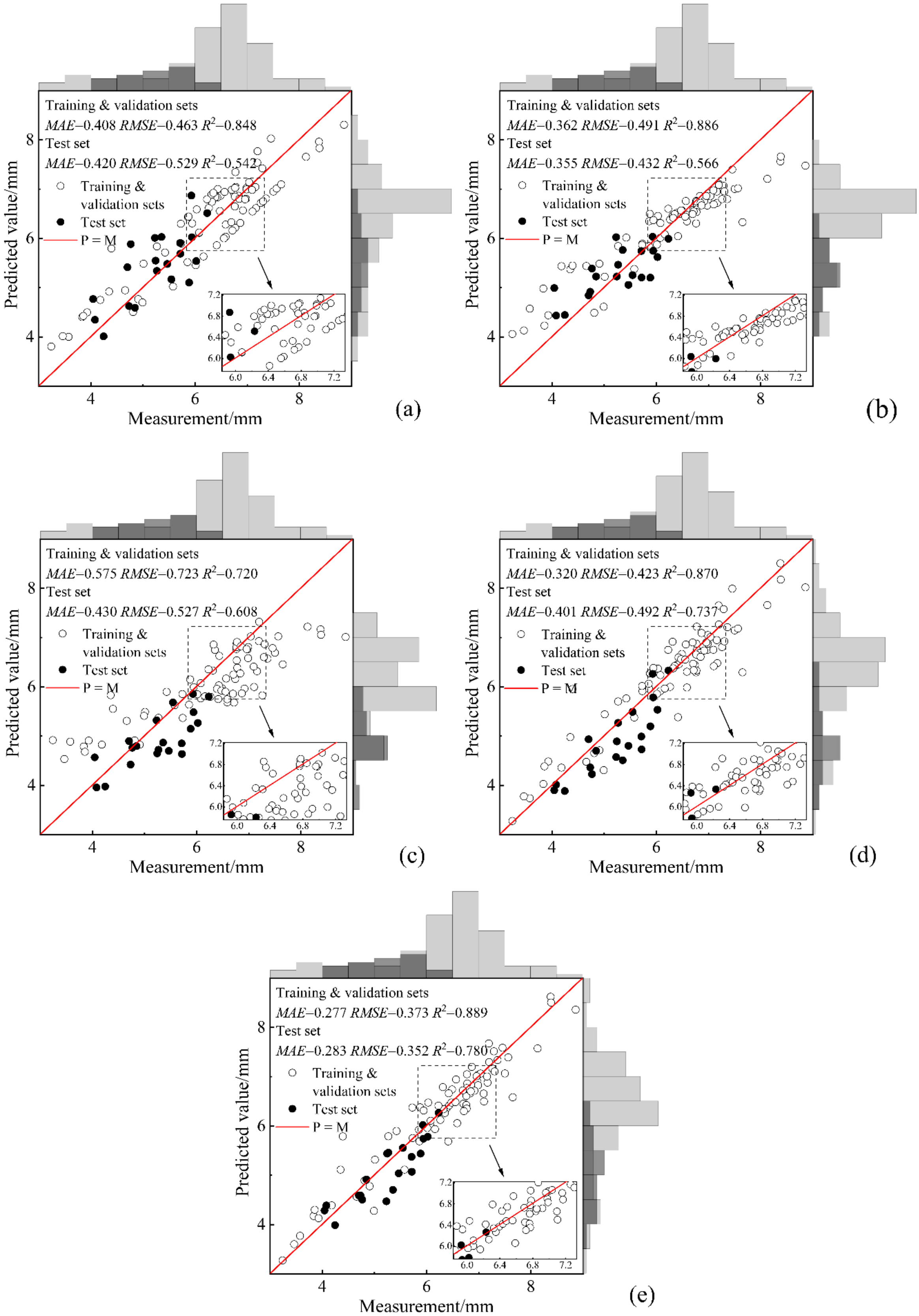
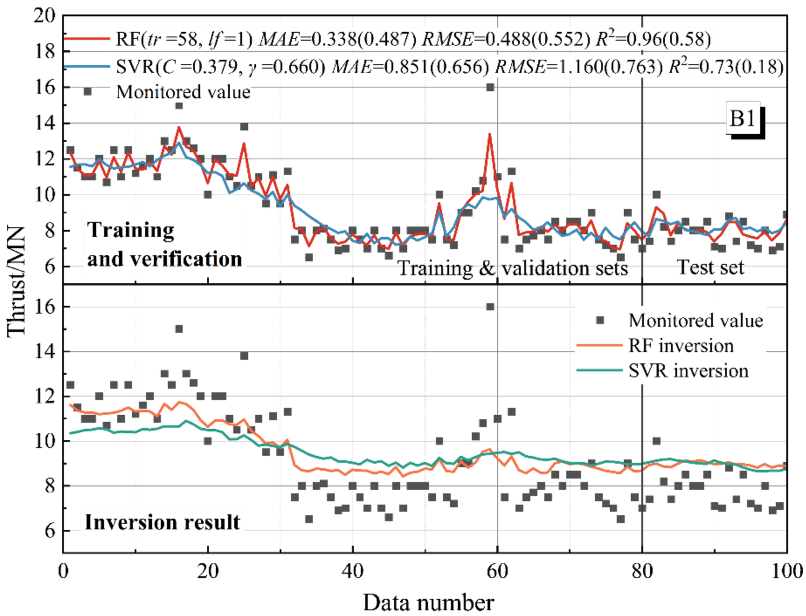


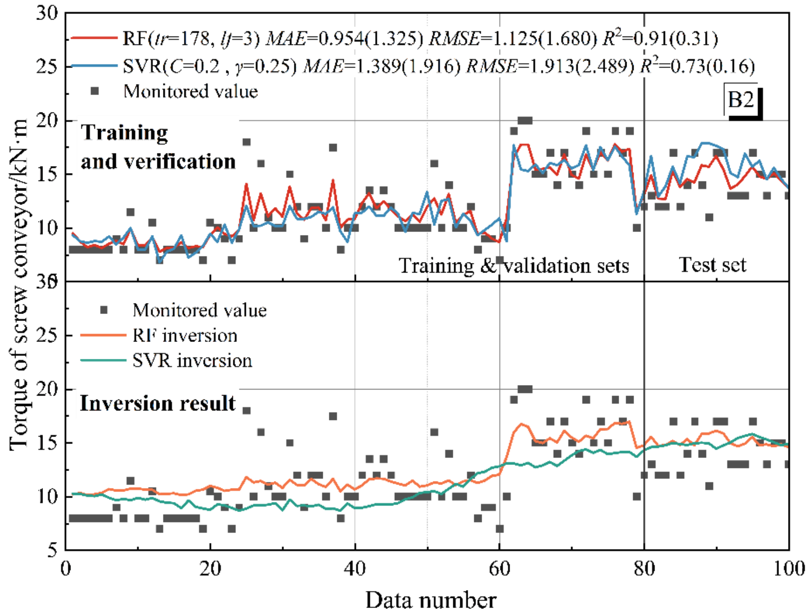
Fig. S9 Predictive performance of the entire data set: (a) The SVR model; (b) The RF model; (c) The Transformer model; (d) The Bi-LSTM model; (e) The MHSA-Bi-LSTM model

Table S2 Data set parameter range for Models 2 and 3

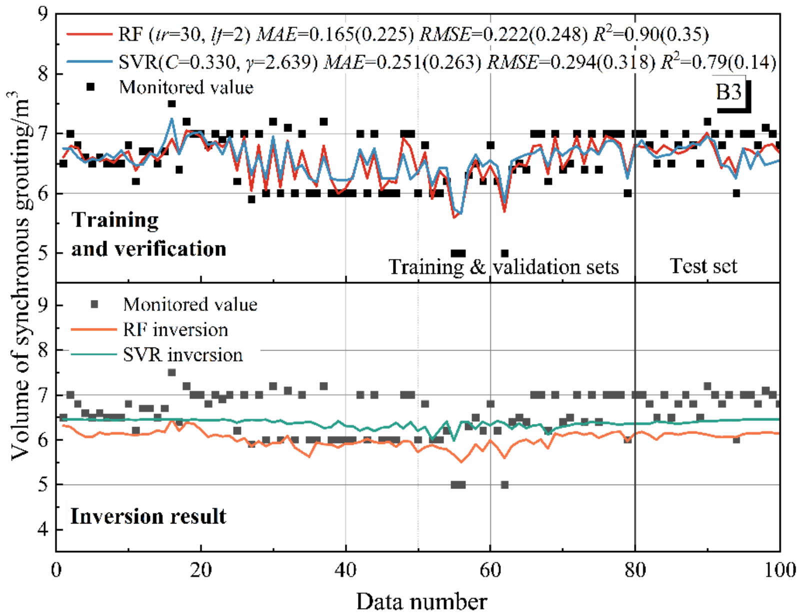
Variable	Parameter type of Model 2 (3)	Model 2 inversion data			Model 3 inversion data			Unit
		Min.	Max.	Ave.	Min.	Max.	Ave.	
R_C	Input (Input)	1.5	-	-	1.5	-	-	rpm
T_C	Input (Input)	1.4	-	-	1.4	-	-	MN·m
P_S	Input (Input)	-	1.5	-	-	1.5	-	bar
V_E	Input (Input)	63.09	-	-	63.09	-	-	m^3
E_{oed}	Input (Input)	6.80	9.48	8.75	6.80	9.48	8.75	MPa
c	Input (Input)	35.67	50.48	46.89	35.67	50.48	46.89	kPa
φ	Input (Input)	21.78	24.66	24.02	21.78	24.66	24.02	°
K_0	Input (Input)	0.335	0.389	0.346	0.335	0.389	0.346	-
D_C	Input (Input)	9.45	12.09	10.82	9.45	12.09	10.82	m
L_W	Input (Input)	3.25	5.50	4.461	3.25	5.50	4.461	m
$S_{PS7.5}$	Input (Input)	-1.96	1.79	0.112	-1.96	1.79	0.112	mm
S_{PS0}	Input (Input)	-1.42	3.07	1.084	-1.42	3.07	1.084	mm
S_S	Input (Input)	1.44	5.49	3.57	1.44	5.49	3.57	mm
T_T	Output (Input)	-	-	-	Model 2 Results			MN
T_{SC}	Output (Input)	-	-	-	Model 2 Results			kN·m
V_G	Output (Input)	-	-	-	Model 2 Results			m^3
R_{SC}	/ (Output)	-	-	-	-	-	-	rpm
S_A	/ (Output)	-	-	-	-	-	-	mm/min



(a) B1: T_T

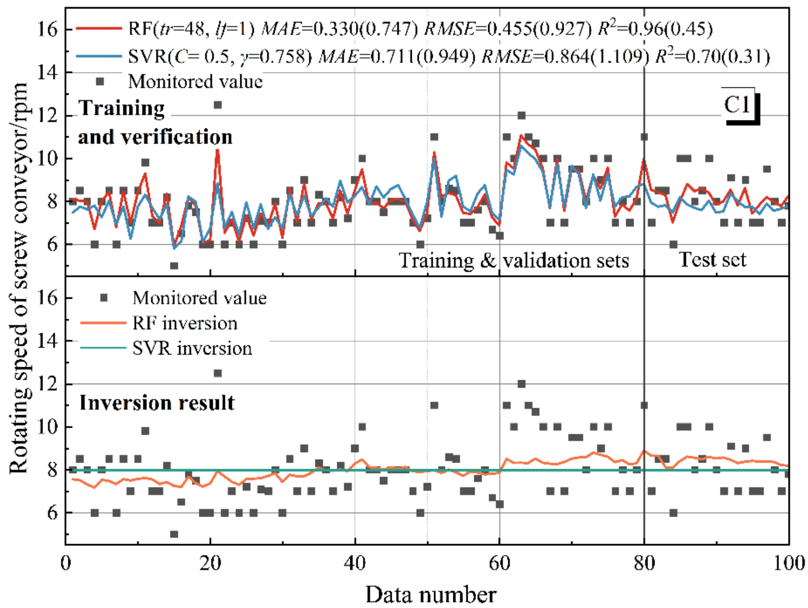


(b) B2: T_{SC}

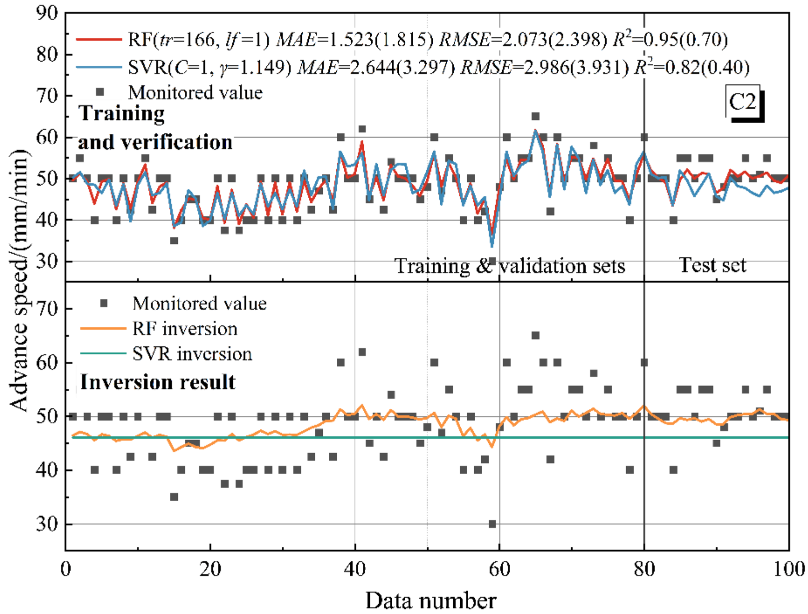


(c) B3: V_G

Fig. S10 Prediction and inversion results of Category B parameters



(a) C1: R_{SC}



(b) C2: S_A

Fig. S11 Prediction and inversion results of Category C parameters

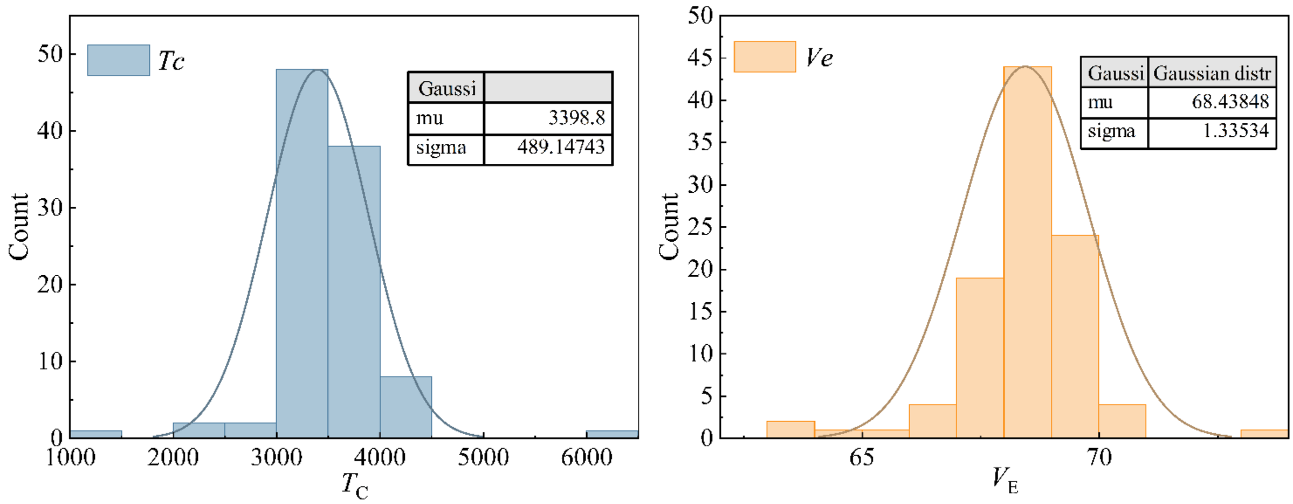


Fig. S12 Initial data distributions of T_C and V_E with fitted Gaussian curves (see also the diagonal histograms in Fig. 9)

References

Breiman L, 1996. Bagging predictors. *Machine Learning*, 24(2):123-140. http://doi.org/10.1007/BF00058655

Eberhart R and Kennedy J, 1995. A new optimizer using particle swarm theory. MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science, Nagoya. IEEE, p. 39-43. http://doi.org/10.1109/MHS.1995.494215

Hu M, Li W, Yan K, et al., 2019. Modern Machine Learning Techniques for Univariate Tunnel Settlement Forecasting: A Comparative Study. *Mathematical Problems in Engineering*, 2019:7057612. http://doi.org/10.1155/2019/7057612

Tin Kam H, 1998. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832-844. http://doi.org/10.1109/34.709601