energies

MDPI

Article

Learning-Aided Optimal Power Flow Based Fast Total Transfer Capability Calculation

Ji'ang Liu [1,*], Youbo Liu [1], Gao Qiu [1,*] and Xiao Shao [2]

[1] College of Electrical Engineering Technology, Sichuan University, Chengdu 610065, China; liuyoubo@scu.edu.cn
[2] State Grid Tianfu New Area Electric Power Supply Company, Chengdu 610041, China; shaoxiao@163.com
* Correspondence: ymm_liujiang@163.com (J.L.); qiugscu@scu.edu.cn (G.Q.)

Abstract: Total transfer capability (TTC) is a vital security indicator for power exchange among areas. It characterizes time-variants and transient stability dynamics, and thus is challenging to evaluate efficiently, which can jeopardize operational safety. A leaning-aided optimal power flow method is proposed to handle the above challenges. At the outset, deep learning (DL) is utilized to globally establish real-time transient stability estimators in parametric space, such that the dimensionality of dynamic simulators can be reduced. The computationally intensive transient stability constraints in TTC calculation and their sensitivities are therewith converted into fast forward and backward processes. The DL-aided constrained model is finally solved by nonlinear programming. The numerical results on the modified IEEE 39-bus system demonstrate that the proposed method outperforms several model-based methods in accuracy and efficiency.

Keywords: total transfer capability; surrogate assisted method; transient stability; deep learning; interior point method

Citation: Liu, J.; Liu, Y.; Qiu, G.; Shao, X. Learning-Aided Optimal Power Flow Based Fast Total Transfer Capability Calculation. *Energies* **2022**, *15*, 1320. https://doi.org/10.3390/en15041320

Academic Editors: Luis Hernández-Callejo, Sergio Nesmachnow and Sara Gallardo Saavedra

Received: 30 December 2021
Accepted: 7 February 2022
Published: 11 February 2022

1. Introduction

Power systems are currently operated near their stability boundary with the significant proliferation of interconnected grids and renewable penetration [1]. Therefore, online monitoring to transfer security margin of inter-area power transfer is in urgent demand. In the electric industry, total transfer capability (TTC), defined as maximum power exchange allowed to withstand multifarious security contingencies, is a widespread metric to quantify such a security margin. Limited by this issue, dispatchers generally use a conservative constant of offline TTC to decide online operations. Undoubtedly, such TTC values can incur the unwanted waste of line capacity and incorrect estimation to security margin. To untie these knots, the essence is to accelerate TTC calculation.

Thus far, several approaches have been proposed to model TTC calculation [2–4]. Among them, methods with only steady-state considered are inapplicable for TTC evaluation involving transient stability (TS) [5]. To enable TS assessment (TSA), TTC is preferred to be modeled as TS constrained (TSC) programming problem. As the models shown in [6–10], differential-algebraic equations (DAEs) representing system dynamics and TS constraints are discretized throughout the time domain simulation period. And the resulting differential equations are incorporated into the optimal power flow (OPF) model. Nevertheless, as mentioned before, solving such models is quite computationally expensive due to the high-dimensional and nonlinear DAEs involved. In light of this, under current time-varying power grids, inefficient physics-dominated methods can be problematic for fast TTC monitors.

Data-driven approaches have become mainstream to increase calculation speed for security assessment in large-scale power systems [11–13]. Reference [11] proposed an online measurement-based TTC estimator using the nonparametric estimation. Sun et al. developed an automatic learning technique based on the linear least-squares fitting method

to extract the TTC operating rules [12]. Unfortunately, these methods own two critical drawbacks. One is that they are hard to capture nonlinear patterns. An empirical and heuristic TTC calculation in the stage of prior sample production is the other problem. It cannot ensure finding the most extreme operating conditions, leading to low fidelity against true modes.

To overcome the first drawback, machine learning (ML) is a promising alternative thanks to its strong nonlinearity learning ability. Reference [13] introduced a hierarchical deep learning machine (HDLM) to successfully achieve real-time TSA, won over other physics-based methods with respect to speed, and beat linear data-driven methods on precision. But sustainable energy is under-investigated. In [14], a TSA framework based on a long short-term memory network was proposed; it improved assessment accuracy by learning from post-fault temporal PMU data dependencies. These applications manifest that ML is a better choice than linear learning methods in nonlinearity modeling tasks. A comparison table with the advantages and disadvantages of the above references is listed in Table 1.

Table 1. A comparison table with the advantages and disadvantages of each reference.

References	Type	Advantages	Disadvantages
[2–4]	Physical-driven model	Focus on steady-state; easy to solve	Transient stability is out of consideration
[6–10]	Physical-driven model with transient stability constraints	Involved transient stability constraints	Computationally expensive
[11–13]	Data-driven model	Faster calculation speed	Hard to capture nonlinear patterns; or sustainable energy is out of consideration

On the other hand, ML can substitute the most time-consuming TSA modules and partially participate in TTC calculation to deal with the second deficiency. This idea follows the classical roadmap of using optimal power flow (OPF) to approach extreme operations but tactfully bypasses high-dimensional modeling such that optimizers can quickly solve TTC. It is technically termed as a learning-aided (also known as surrogate-assisted) method (LAM) [15–18], which utilizes ML algorithms to surrogate the most complex and computationally intensive parts in optimization problems. Reference [18] proposed a method that makes a fusion between surrogates and the evolutionary algorithm to improve the efficiency of optimizing high-dimensional expensive problems. In [1], LAM is also utilized to solve the TTC constrained operation planning problem. The above studies show that LAM can speed up solving optimization problems. At the same time, because it is a data-mechanism hybrid-driven method rather than an utterly data-driven method, it performs better in terms of fidelity.

By prioritizing both merits of physics- and data-driven modeling, this paper proposed a learning-aided optimal power flow based fast TTC calculation methods with the following features:

Deep belief network (DBN) is advocated to surrogate computationally intensive and high-dimensional time-domain based transient stability modelling. This learning-aided scheme allows us to significantly reduce complexity of TTC calculation.

- DBN backwards process is conducted to derive sensitivity of transient stability margin. This sensitivity supports fast and accurate decision for the most extreme growth path of generation and load. The TTC solved under such path is conservative and robust to account for a reliable security indicator.
- Thanks to the above merits, interior point method (IPM) is then introduced to fast calculate TTC. Specifically, DBN forwards and backwards processes respectively provide fast and accurate transient stability inference and gradient information for

IPM. This scheme is firstly used in OPF-based TTC calculation, and numerical studies justified its merits of compromising calculation efficiency and accuracy.

- A comprehensive comparative study is constructed. Numbers of traditional methods, such as the TSCOPF method [9], the sensitivity-based method [19], the repeated power flow (RPF) method [20], and the direct data-driven method [21], are used to demonstrate the superiority of our method.

The organization of this paper is as follows: Section 2 introduces TS constrained optimal power flow (TSCOPF), adopted to model TTC calculation. The learning-aided model for the TS constraints is introduced in Section 3. Section 4 details the proposed solving scheme method, where the Jacobin and Hessian matrices of the learning model are deduced to analytical form to enable combination with nonlinear programming. Section 5 illustrates the numerical study. Finally, the conclusion is presented in Section 6.

2. TTC Calculation with TSCOPF

We believe the OPF method is a brilliant choice because the optimization procedure enables a theoretical search for extreme operating conditions representing TTC. Therefore, TSCOPF is adopted to model TTC calculation problem in this section. According to [22], the generic OPF method for calculating TTC can be formulated as follows:

$$\begin{array}{c} \text{Max } f(\boldsymbol{y},\boldsymbol{u}) \\ \text{s.t. } g(\boldsymbol{y},\boldsymbol{u}) = 0 \\ h(\boldsymbol{y},\boldsymbol{u}) \le 0 \end{array} \tag{1}$$

where $\boldsymbol{y}$,$\boldsymbol{u}$ are the state and control variable vector of the system; and $g(\cdot)$, $h(\cdot)$ are the set of equality and inequality constraints, respectively.

(1) Objective function: It aims to maximize the sum of the active power output of all generators in the source area, i.e.,

$$\text{Max } f(\boldsymbol{y},\boldsymbol{u}) = \sum\nolimits_{k \in Ssou} P_{Gk}, \tag{2}$$

where P_{Gk} is generator active power output at bus k; and S_{sou} means the source area bus set.

(2) Static equality constraints: Power flow equations are formed under polar coordinates, shown below:

$$\begin{array}{l} P_{Gi} - P_{Di} - V_i \sum\nolimits_{j=1}^{n} V_j (G_{ij}\cos\theta_{ij} + B_{ij}\sin\theta_{ij}) = 0, \\ Q_{Gi} - Q_{Di} - V_i \sum\nolimits_{j=1}^{n} V_j (G_{ij}\sin\theta_{ij} - B_{ij}\cos\theta_{ij}) = 0 \end{array} \tag{3}$$

where P_{Gi}, P_{Di} represent active generation and demand for bus i; Q_{Gi}, Q_{Di} are reactive generation and demand for bus i, respective; V_i and θ_i are the voltage magnitude and phase angle of bus i, and $\theta_{ij} = \theta_i - \theta_j$; $G_{ij} + jB_{ij}$ is the driving point admittance and the transfer admittance; n is the number of buses.

(3) Static inequality constraints:

$$\begin{array}{c} {P_{Gi}}^{\min} \le P_{Gi} \le {P_{Gi}}^{\max}, Gi \in S_G \cup S_W \\ {Q_{Gi}}^{\min} \le Q_{Gi} \le {Q_{Gi}}^{\max}, Gi \in S_G \cup S_W \\ {V_i}^{\min} \le V_i \le {V_i}^{\max}, i \in S_n \\ P_{ij} \le {P_{ij}}^{\max}, ij \in S_l \end{array} \tag{4}$$

where ${P_{Gi}}^{\min}$, ${P_{Gi}}^{\max}$, ${Q_{Gi}}^{\min}$, ${Q_{Gi}}^{\max}$ are the lower and upper limits of the generator active and reactive power at bus k, respective; ${V_i}^{\min}$ and ${V_i}^{\max}$ are the lower and upper limits of the voltage at bus i; ${P_{ij}}^{\max}$ is the transmission threshold of line ij; S_G, S_W, S_n, S_l are the sets of generators, wind farms, buses, and lines.

(4) Transient stability constraints: This paper adopts the classical generator model to analyze transient stability. During the dynamic process, loads are modeled as constant impedance. Hence, generic TS models can be simplified as follows:

$$\begin{gathered} \boldsymbol{x}'(t) = \rho_c(\boldsymbol{x}(\mathrm{t}), \boldsymbol{y}(t), \boldsymbol{u}), \\ \psi_c(\boldsymbol{x}(t), \boldsymbol{y}(t), \boldsymbol{u}) \geq 0, c \in S_c, t \in (t_0, t_{\mathrm{end}}] \end{gathered} \tag{5}$$

where $\boldsymbol{x}$, $\boldsymbol{y}$ are the algebraic and state variables; $[\boldsymbol{x}(t), \boldsymbol{y}(\mathrm{t})]$ refers to the operating condition during the transient period $(t_0, t_{\mathrm{end}}]$; S_c is a set of pre-contingencies; $\psi_c(\cdot)$ is the transient stability criterion used in this paper [23], and it is shown as follows:

$$\begin{gathered} |\, \delta_i(t) - \delta_{COI}(t) \,| \leq \delta_{thr}, t \in (t_0, t_{\mathrm{end}}] \\ \delta_{COI}(t) = (\textstyle\sum_i M_i \cdot \delta_i(t)) \,/\, (\sum_i M_i), i \in \{1, \ldots, n_G\} \end{gathered} \tag{6}$$

where $\delta_i(t)$ is the rotor angles of generator i; $\delta_{COI}(t)$ is the rotor angle under the center of inertia (COI); δ_{thr} is the instability threshold that is usually set as 180 degree [23]; M_i represents the inertia constant of the ith generator; and n_G denotes the number of generators.

It should be mentioned the DAEs Equation (5) encompasses numerous time-domain variables, i.e., $\delta_i(t)$. With more precise timestep and more contingencies to be checked, the dimensionality of Equation (5) will be of exponential growth.

3. Proposed Surrogate Model

In TSC programming problem, exact state and parameter estimation for dynamic components (e.g., synchronous generators) must be conducted to truly model the transient process. This is difficult because large-scale state estimation is challenging regarding efficiency and precision. A sensible alternative is to directly encapsulate transient stability dynamics in a parameterized model so as to bypass state estimation. Rich data is needed, fortunately, it can be easily gathered nowadays in smart grid.

As reported before, TSA significantly increases the computational burden of solving the OPF model. To reduce the massive time-domain variables, a data-driven learning-aided model is proposed. This model allows us to map Equation (5) into a parametric space, such that the time-domain variables can be surrogated by ML structural parameters independent of optimization, and few parallel forwards processes of ML are enabled to circumvent quantities of DAEs.

3.1. Data Sample Generation

The first step of training such learning-aided models is data generation. To this end, random operations are sampled and simulated under prior distributions of power systems. To simplify illustrations, we respectively denote the input features and the target features as $\boldsymbol{X}$ and $\boldsymbol{Y}$. $\boldsymbol{X}$ covers almost all variables that SCADA can measure, while $\boldsymbol{Y}$ is the TS margin index. Equation (7) details the data structure:

$$\begin{gathered} \boldsymbol{X} = \{\boldsymbol{P}_G, \boldsymbol{V}_G, \boldsymbol{P}_D, \boldsymbol{Q}_D, \boldsymbol{V}_b\}, G \in S_G \cup S_W, D \in S_D \\ \boldsymbol{Y} = \{\Gamma_c\}, c \in S_c \end{gathered} \tag{7}$$

where $\boldsymbol{P}_G$ and $\boldsymbol{P}_D$ are the characteristic vector of active generator output and active load, respectively; $\boldsymbol{Q}_D$ is the vector of reactive load; $\boldsymbol{V}_G$ represents the voltage of buses where generators are located; and $\boldsymbol{V}_b$ means the voltage of other buses. Γ_c represents the TS margin of the corresponding operation. In this paper, TS index (TSI) is adopted to quantify TS margin, which can be formulated as:

$$\begin{gathered} \mathrm{TSI} = 100 \times (\delta_{thr} - |\, \delta_{\max} \,|)/(\delta_{thr} + |\, \delta_{\max} \,|), \\ \delta_{\max} = \max(|\, \delta_{Gi} - \delta_{Gj} \,|), Gi, Gj \in S_G \end{gathered} \tag{8}$$

where $\delta_{\max}$ is the maximum power angle difference during the post-fault duration.

Now turning to introduce calculation for Equation (7). To attain $\boldsymbol{X}$, we firstly sample controllable variables under their prior limits by Equation (9), and loads under historical distributions by Equation (9):

$$\begin{array}{l} \boldsymbol{X}_{gen} = \{\boldsymbol{X}^1{}_{gen}; \ldots ; \boldsymbol{X}^n{}_{gen}\} = \{\boldsymbol{P}^1{}_G, \boldsymbol{V}^1{}_G; \ldots ; \boldsymbol{P}^n{}_G, \boldsymbol{V}^n{}_G\}, \\ \boldsymbol{X}_{load} = \{\boldsymbol{X}^1{}_{load}; \ldots ; \boldsymbol{X}^n{}_{load}\} = \{\boldsymbol{P}^1{}_D, \boldsymbol{Q}^1{}_D; \ldots ; \boldsymbol{P}^n{}_D, \boldsymbol{Q}^n{}_D\} \end{array} \quad (9)$$

where $\boldsymbol{X}_{gen}$, $\boldsymbol{X}_{load}$ are the control and load variables subsets of $\boldsymbol{X}$, respectively; n is the number of samples.

The power flow program is then performed to get equilibrium points to determine the state variables $\boldsymbol{V}_b$. Notably, samples should be evenly distributed over operational space to ensure the generalization ability of the learning-aided model. Therefore, Latin hypercube sampling (LHS) is adopted to generate samples in this paper [24].

Afterward, we impose disturbances in contingencies for one equilibrium point of $\boldsymbol{X}$ to obtain post-fault trajectories to compute TS margin Γ_c. Via traversing each point in $\boldsymbol{X}$, $\boldsymbol{Y}$ can be collected. Supervised learning can herewith be utilized to learn the learning-aided model.

3.2. Deep Belief Network Based TSA Learning-Aided Model

According to the data structure, the deep belief network (DBN) is an advisable alternative for our goal. DBN is a probability generation model that stacks multiple restricted Boltzmann machines (RBMs) and a fully connected layer. RBM is an unsupervised network composed of a visible and hidden layer, and it can probabilistically reconstruct input features by two-way connections between the two layers.

As an energy-based model, the energy function of RBM is calculated by [25]:

$$E(\boldsymbol{v},\boldsymbol{h}) = -\boldsymbol{v}^T\boldsymbol{w}\boldsymbol{h} - \boldsymbol{a}^T\boldsymbol{v} - \boldsymbol{b}^T\boldsymbol{h}, \boldsymbol{w}\in \mathrm{R}^{nh\times nv}, \boldsymbol{a}\in \mathrm{R}^{nv}, \boldsymbol{b}\in \mathrm{R}^{nh} \quad (10)$$

where $\boldsymbol{v}$,$\boldsymbol{h}$ are the visible and hidden layer matrices; $\boldsymbol{a}$,$\boldsymbol{b}$ are the bias matrices of $\boldsymbol{v}$,$\boldsymbol{h}$ respectively; and $\boldsymbol{w}$ is the weight matrix between two layers. The joint probability distribution $P(\boldsymbol{v},\boldsymbol{h})$ of $\boldsymbol{v}$ and $\boldsymbol{h}$ is formulated by:

$$P(\boldsymbol{v},\boldsymbol{h}) = Z^{-1}\, e^{-E(\boldsymbol{v},\boldsymbol{h})}, Z = \sum\nolimits_{\boldsymbol{v},\boldsymbol{h}} e^{-E(\boldsymbol{v},\boldsymbol{h})} \quad (11)$$

where Z is the normalization factor that ensures the sum of the probability distribution is 1. The marginal probability of $\boldsymbol{v}$ and $\boldsymbol{h}$, which are also called the likelihood functions, can be formulated as:

$$P(\boldsymbol{v}) = Z^{-1}\sum\nolimits_{\boldsymbol{h}} e^{-E(\boldsymbol{v},\boldsymbol{h})}, P(\boldsymbol{h}) = Z^{-1}\sum\nolimits_{\boldsymbol{v}} e^{-E(\boldsymbol{v},\boldsymbol{h})} \quad (12)$$

Due to the lack of intra-layer connections in RBM, the activations of units in the visible and hidden layers are independent. Therefore, when the visible layer (or hidden layer) units state is given, we can deduce the formulation of the conditional probability that an individual unit of the hidden layer (or visible layer) is activated as:

$$\begin{array}{l} P(h_i = 1 \mid \boldsymbol{v}) = M(b_i + \sum_i w_{ij} \cdot v_j), h_i\in\boldsymbol{h}, v_j\in\boldsymbol{v}, w_{ij}\in\boldsymbol{w} \\ P(v_i = 1 \mid \boldsymbol{h}) = M(a_j + \sum_j w_{ij} \cdot h_i), a_j\in\boldsymbol{a}\in \mathrm{R}^{nv}, b_i\in\boldsymbol{b}\in \mathrm{R}^{nh} \end{array} \quad (13)$$

where $M(\cdot)$ is the activation function, and in the paper, it is the *Sigmoid* function. Then, the conditional probability of $\boldsymbol{h}$ (or $\boldsymbol{v}$) given $\boldsymbol{v}$ (or $\boldsymbol{h}$) can be obtained:

$$P(\boldsymbol{h} \mid \boldsymbol{v}) = \prod\nolimits^{nh}{}_{i=1} P(h_i \mid \boldsymbol{v}), P(\boldsymbol{v} \mid \boldsymbol{h}) = \prod\nolimits^{nv}{}_{j=1} P(v_j \mid \boldsymbol{h}), h_i\in\boldsymbol{h}, v_j\in\boldsymbol{v} \quad (14)$$

where nh,nv are the number of units in the hidden and visible layer, respectively.

Training RBM is to maximize the following likelihood L:

$$\ln L = \ln\prod\nolimits_{\boldsymbol{v}\in Strain} P(\boldsymbol{v}), \quad (15)$$

where S_{train} is the training sample set. The commonly used numerical method for maximizing (15) is gradient ascent, which iteratively updates the parameters. Take $\boldsymbol{w}$ as an example, and the weight w_{ij} is updated via Equations (16) and (17):

$$w_{ij} = w_{ij} + \eta \cdot (\partial \ln(P(\boldsymbol{v})))/(\partial w_{ij}),\ w_{ij} \in \boldsymbol{w} \tag{16}$$

$$(\partial \ln(P(\boldsymbol{v})))/(\partial w_{ij}) = P(h_i = 1 \mid \boldsymbol{v})v_j - \sum_v P(\boldsymbol{v})P(h_i = 1 \mid \boldsymbol{v})v_j,\ h_i \in \boldsymbol{h},\ v_j \in \boldsymbol{v} \tag{17}$$

where η is the learning rate.

DBN training consists of two parts: pre-training and fine-tuning. In the pre-training part, any two connected layers except the fully connected layer can be regarded as an RBM. These RBMs are trained to obtain better initial weights and to alleviate the gradient disappearance problem. In the fine-tuning part, the trained RBMs are connected with the fully connected layer. The sample sets $[\boldsymbol{X}, \boldsymbol{Y}]$ and the global learning algorithm are then used for supervised fine-tuning of the DBN, learning the mapping between input data and labels. Thence, the mathematical model of an l-layer DBN can be simplified by Equation (18):

$$\Psi(\boldsymbol{X}) = O(M(D_{l-1}(\ldots\ M(D_1(\boldsymbol{X}))\ \ldots\ ))), \tag{18}$$

$$D_i(\boldsymbol{x}_i) = \boldsymbol{w}_i\boldsymbol{x}_i + \boldsymbol{b}_i,\ w_i \in \mathrm{R}^{ni \times n(i-1)},\ i = 1, \ldots, l-1 \tag{19}$$

$$\begin{gathered} \boldsymbol{w}_i = [\boldsymbol{w}_i{}^1, \ldots, \boldsymbol{w}_i{}^{ni}],\ \boldsymbol{w}_i{}^{ni} \in \mathrm{R}^{1 \times n(i-1)} \\ \boldsymbol{b}_i = [b_i{}^1, \ldots, b_i{}^{ni}] \end{gathered} \tag{20}$$

where $O(\cdot)$ is the output function of the fully connected layer, and $O(\boldsymbol{x}) = D_l(\boldsymbol{x}_l)$. The loss function can be defined as the weighted sum of the estimated error and L_2 norm, i.e.:

$$\text{Min } \alpha\|\boldsymbol{Y} - \Psi(\boldsymbol{X})\|_2{}^2 + (1-\alpha)\sum_{i=1}^{n} \|\boldsymbol{w}_i\|_2{}^2, \tag{21}$$

Equation (21) can be solved by training and fine-tuning the DBN model [24].

After training, the learning-aided model is reformed as Equation (22):

$$\Gamma_c = \Psi_c(\boldsymbol{X}),\ c \in S_c \tag{22}$$

3.3. Learning-Aided OPF for TTC Calculation

The trained learning-aided model is finally forwarded to replace (5)~(6) to mitigate the TTC computational burden. The reformed learning-aided OPF for calculating TTC is given as follows:

$$\begin{gathered} \text{Maximize (2)} \\ \text{s.t. (3)}\sim\text{(4)} \\ \Gamma_c \geq 0,\ c \in S_c \end{gathered} \tag{23}$$

In Equation (23), steady-state physics remains the same, but dynamics become a data model. This modeling strategy possesses several merits: (1) it remarkably reduces the solving complexity of the full physics version. (2) it preserves physics to decrease adverse effects from significant learning errors. A common way to solve Equation (23) is gradient-free algorithms [26–28]. However, these algorithms characterize cumbersome stochastic search mechanism. A fast-solving algorithm for such physics and data hybrid model is still under exploitation.

4. Proposed Solution Method

In this paper, the interior point method (IPM) [29] is conducted to solve (25). Towards this end, the *Jacobian* and *Hessian* matrix of the trained DBN model is analyzed.

4.1. Interior Point Method

For the sake of simplification, model Equation (23) is firstly reformed as the following canonical form:

$$\begin{gathered} \text{Max } F(\boldsymbol{x}), \\ \text{s.t. } \boldsymbol{G}(\boldsymbol{x}) = \boldsymbol{0}, \\ [\boldsymbol{H}(\boldsymbol{x}) \leq \boldsymbol{0}] = [\boldsymbol{H}_c(\boldsymbol{x}) \leq \boldsymbol{0}, \boldsymbol{H}_S(\boldsymbol{x}) \leq \boldsymbol{0}] \end{gathered} \tag{24}$$

where $F(\boldsymbol{x})$ is the objective function; $\boldsymbol{G}(\boldsymbol{x}) = [G_1(\boldsymbol{x}), \ldots, G_{\mathrm{m}}(\boldsymbol{x})]^T$ is the nonlinear equality constraints; $\boldsymbol{H}(\boldsymbol{x}) = [H_1(\boldsymbol{x}), \ldots, H_{\mathrm{r}}(\boldsymbol{x})]^T$ is the non-linear inequality constraints, and $\boldsymbol{H}_c(\boldsymbol{x})$, $\boldsymbol{H}_S(\boldsymbol{x})$ are the constraints in (4) and the surrogate model in (23b), respectively; r,m are the number of inequality and equality constraints.

Use IPM to solve Equation (24), and the steps are as follows [29]:

1. Add slack variables $\boldsymbol{l} = [l_1, \ldots, l_r]^T$ ($\boldsymbol{l} > \boldsymbol{0}$) and $\boldsymbol{u} = [u_1, \ldots, u_r]^T$ ($\boldsymbol{u} > \boldsymbol{0}$) to transform $\boldsymbol{H}(\boldsymbol{x})$ into equality constraints;
2. Introduce the disturbance factor μ ($\mu > 0$) to transfer $F(\boldsymbol{x})$ into the barrier function, which makes it impossible for the barrier objective function to find an extremal solution on the boundary, and the optimal solution can only be obtained when the constraints are satisfied;
3. Apply Lagrangian multiplier method to solve the transformed model, and the Lagrangian function is formulated as:

$$\begin{gathered} L = F(\boldsymbol{x}) - \boldsymbol{\zeta}^T \boldsymbol{G}(\boldsymbol{x}) - \boldsymbol{z}^T[\boldsymbol{H}(\boldsymbol{x}) - \boldsymbol{l} - \boldsymbol{H}_{\min}] - \boldsymbol{\omega}^T[\boldsymbol{H}(\boldsymbol{x}) + \boldsymbol{u} - \boldsymbol{H}_{\max}] \\ - \mu \sum_{i=1}^{r} \log(l_i) - \mu \sum_{i=1}^{r} \log(l_i), \end{gathered} \tag{25}$$

   where $\boldsymbol{\zeta}$,$\boldsymbol{z}$,$\boldsymbol{\omega}$, are Lagrangian multipliers, respectively.
4. Calculate μ via Equation (26):

$$\mu = \sigma(\boldsymbol{l}^T \boldsymbol{z} - \boldsymbol{u}^T \boldsymbol{\omega})/2\mathrm{r}, \tag{26}$$

   where σ denotes the central parameter.
5. Consider the Karush–Kuhn–Tucker (KKT) conditions and adopt the Newton method, the matrix form of the modified equations can be deduced as:

$$\begin{gathered} \boldsymbol{\Lambda} \cdot \Delta \boldsymbol{x} + (\partial \boldsymbol{G}(\boldsymbol{x})/\partial \boldsymbol{x}) \cdot \Delta \boldsymbol{x} = \boldsymbol{\Phi}, \\ (\partial \boldsymbol{G}(\boldsymbol{x})^T/\partial \boldsymbol{x}) \cdot \Delta \boldsymbol{x} = \boldsymbol{G}, \\ \boldsymbol{\Lambda} = (\partial^2 \boldsymbol{G}(\boldsymbol{x})/\partial \boldsymbol{x}^2)\boldsymbol{\zeta} + (\partial^2 \boldsymbol{H}(\boldsymbol{x})/\partial \boldsymbol{x}^2)(\boldsymbol{z} + \boldsymbol{\omega}) - (\partial^2 F(\boldsymbol{x})/\partial \boldsymbol{x}^2) \\ + (\partial \boldsymbol{H}(\boldsymbol{x})/\partial \boldsymbol{x})(\boldsymbol{u}^{-1}\boldsymbol{\omega} - \boldsymbol{l}^{-1}\boldsymbol{z})(\partial \boldsymbol{H}(\boldsymbol{x})/\partial \boldsymbol{x})^T, \\ \boldsymbol{\Phi} = -L_{\boldsymbol{x}} - (\partial \boldsymbol{H}(\boldsymbol{x})/\partial \boldsymbol{x})[\boldsymbol{L}^{-1}(L_{\boldsymbol{l}}{}^{\mu} + \boldsymbol{Z}L_{\boldsymbol{z}}) + \boldsymbol{U}^{-1}(L_{\boldsymbol{u}}{}^{\mu} + \boldsymbol{W}L_{\boldsymbol{\omega}})] \end{gathered} \tag{27}$$

   where $L_{\boldsymbol{x}}$, $L_{\boldsymbol{z}}$, $L_{\boldsymbol{\omega}}$, $L_{\boldsymbol{l}}{}^{\mu}$ and $L_{\boldsymbol{u}}{}^{\mu}$ are the partial derivatives of L to $\boldsymbol{x}$, $\boldsymbol{z}$, $\boldsymbol{\omega}$, $\boldsymbol{l}$ and $\boldsymbol{u}$.

   Besides, the corrections of $\boldsymbol{z}$, $\boldsymbol{\omega}$, $\boldsymbol{l}$ and $\boldsymbol{u}$ can be calculated via (28):

$$\begin{gathered} \Delta \boldsymbol{z} = \boldsymbol{L}^{-1} L_{\boldsymbol{l}}{}^{\mu} - \boldsymbol{L}^{-1}\boldsymbol{Z}\Delta \boldsymbol{l}, \\ \Delta \boldsymbol{l} = (\partial \boldsymbol{H}(\boldsymbol{x})/\partial \boldsymbol{x})^T \Delta \boldsymbol{x} - L_{\boldsymbol{z}}, \\ \Delta \boldsymbol{\omega} = \boldsymbol{U}^{-1} L_{\boldsymbol{u}}{}^{\mu} - \boldsymbol{U}^{-1}\boldsymbol{W}\Delta \boldsymbol{u}, \\ \Delta \boldsymbol{u} = -(\partial \boldsymbol{H}(\boldsymbol{x})/\partial \boldsymbol{x})^T \Delta \boldsymbol{x} + L_{\boldsymbol{\omega}} \end{gathered} \tag{28}$$

   where $\boldsymbol{Z} = \mathrm{diag}(\boldsymbol{z})$; $\boldsymbol{W} = \mathrm{diag}(\boldsymbol{\omega})$; $\boldsymbol{L} = \mathrm{diag}(\boldsymbol{l})$; and $\boldsymbol{U} = \mathrm{diag}(\boldsymbol{u})$.
6. Use the corrections calculated via Equations (27) and (28) to update the variables as follows:

$$\begin{gathered} \boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \alpha_p \Delta \boldsymbol{x}, \ \boldsymbol{\zeta}^{(k+1)} = \boldsymbol{\zeta}^{(k)} + \alpha_d \Delta \boldsymbol{\zeta}, \\ \boldsymbol{l}^{(k+1)} = \boldsymbol{l}^{(k)} + \alpha_p \Delta \boldsymbol{l}, \ \mathbf{z}^{(k+1)} = \mathbf{z}^{(k)} + \alpha_d \Delta \mathbf{z}, \\ \boldsymbol{u}^{(k+1)} = \boldsymbol{u}^{(k)} + \alpha_p \Delta \boldsymbol{u}, \ \boldsymbol{\omega}^{(k+1)} = \boldsymbol{\omega}^{(k)} + \alpha_d \Delta \boldsymbol{\omega}, \end{gathered} \tag{29}$$

where the step size α_p and α_d for each update are shown in Equation (30).

$$\begin{aligned} \alpha_p &= 0.9995\min[\min(-\boldsymbol{l}_i/\Delta\boldsymbol{l}_i, \boldsymbol{l}_i < 0; -\boldsymbol{l}_i/\Delta\boldsymbol{u}_i, \boldsymbol{u}_i < 0), 1], \\ \alpha_d &= 0.9995\min[\min(-z_i/\Delta z_i, z_i < 0; -\boldsymbol{\omega}_i/\Delta\boldsymbol{\omega}_i, \boldsymbol{\omega}_i > 0), 1], i = 1, \ldots, r \end{aligned} \tag{30}$$

7. Termination condition: if $(\boldsymbol{l}^T\boldsymbol{z} - \boldsymbol{u}^T\boldsymbol{\omega}) < \varepsilon$, the current $\boldsymbol{x}$ is output; else re-execute (4) to (6). Here ε represents the specified threshold.

4.2. Deducing Analytical Surrogate Model for IPM

As shown in Equation (26), the gradient of functions F,$\boldsymbol{G}$ and $\boldsymbol{H}$ are needed in the process of IPM. The gradient of F, $\boldsymbol{G}$ and $\boldsymbol{H}_c$ can be obtained directly. However, since the surrogate model is a "black-box", the gradient of $\boldsymbol{H}_s$ cannot be simply calculated. Based on Equations (22) and (23), the gradient of $\boldsymbol{H}_s$ can be transformed into:

$$\begin{aligned} \bigtriangledown\boldsymbol{H}_s &= \bigtriangledown\Gamma_c = \bigtriangledown\Psi_c(\boldsymbol{X}), c \in S_c \\ \bigtriangledown^2\boldsymbol{H}_s &= \bigtriangledown^2\Gamma_c = \bigtriangledown^2\Psi_c(\boldsymbol{X}) \end{aligned} \tag{31}$$

Next, by DBN backwards process, Equation (31) is deduced to get the Jacobin and Hessian matrix, which are also known as sensitivities of transient stability against optimization variables. See Appendix A for the detailed DBN backwards process. The algorithm flow and implementation of the proposed method are shown in Figure 1.

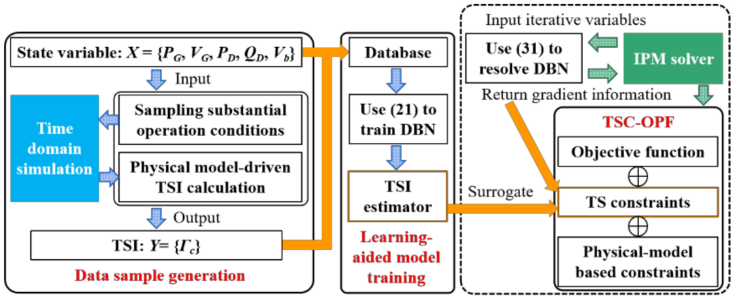


Figure 1. The flow chart of the proposed method.

5. Numerical Case Study

5.1. Test System

The proposed method is testified on the modified IEEE 39-bus system. The base power of the system is 100 MW, and the system is divided into the source (Area I) and sink (Area II) areas by four tie-lines 1–39, 2–3, 3–18, and 16–17, as shown in Figure 2. Two wind farms with a total capacity of 500 MW are connected to buses 17 and 21. As shown in Section 3.1, TSI calculated by power angle is adopted to quantify TS margin. So, this paper assumes that the wind farms have sufficient reactive power reserves and low voltage ride-through capability to ensure they do not trip. A three-phase short circuit on each tie-line is pre-selected as the contingencies.

5.2. Learning-Aided Model Construction

As mentioned before, generation prior distribution is assumed to be a uniform distribution over generators' nominal limits. Regarding generation and load balance constrained, the total load is determined as the sum of generation. Nodal load is then acquired by sampling from historical load distributions. As for the settings of time-domain simulation, fault start time is 0.1 s, simulation period is 2 s, and timestep is 0.05 s. Following the above preconditions, 10,000 samples are generated, of which the ratios to the training set, validation set, and test set are 80%, 10%, and 10%.

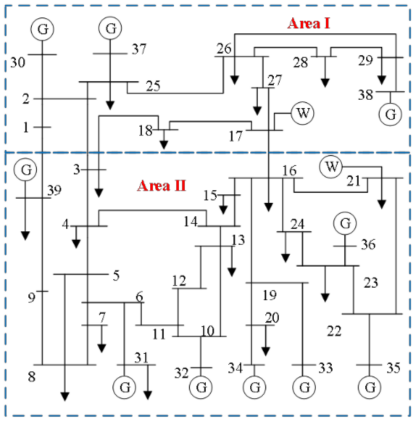
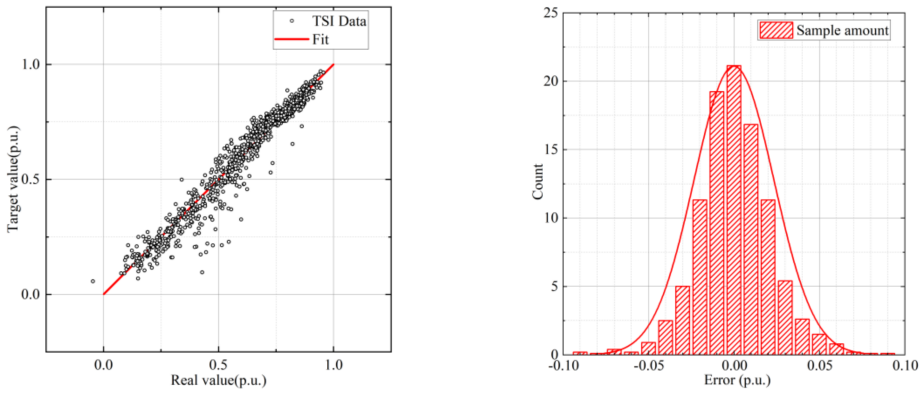


Figure 2. The modified IEEE 39-bus system for case study.

The DBN structure from the input layer to the output layer is {93-40-20-10-5-4}, where the elements stand for neuron quantity. Sigmoid is selected as the activation function. After training the samples, the DBN is forwarded to be tested on out-of-sample sets (i.e., test set). The scatter of estimates vs. actual values and error distribution is shown in Figure 3. The coefficient of determination (R2) is 0.9814, and the mean square error is 5.84×10^{-4}. As shown in Figure 3b, the error distribution approximately obeys a normal distribution, and the mean and standard deviation of the estimation error are 0 and 0.023. It can be found that 95% of the samples are in the interval of $[-0.042, 0.046]$ through statistics, and the error with the 95% confidence level of the normal distribution is 0.004. Figure 3 demonstrates that the proposed learning model can render accurate TSA and strongly generalizes.



(**a**) (**b**)

Figure 3. The visualization of testing the trained learning model: (**a**) estimate vs. true; (**b**) error distribution.

To further verify the performance of DBN, comparisons against back propagation neural network (BPNN), support vector regression (SVR), and regression tree (RT) are carried out, and the outcomes are given in Table 2. Mean square error (MSE) and square correlation coefficient (SCC) are used to evaluate the performance. You can see clearly that the DBNs beat other learning methods; thus, it can be concluded that DBN is the best one in TSA tasks.

Table 2. Accuracy comparison of each TSI surrogate model on test sets.

Indicator	2-Layer DBN	3-Layer DBN	BPNN	SVR	RT
MSE/p.u.	0.0054	0.0019	0.0023	0.0346	0.0927
SCC	0.9480	0.9712	0.9627	0.9171	0.8814

5.3. The Results of TTC Fast Calculation

In this section, the proposed method is testified and compared with other methods, such as the TSC-OPF method [9], the sensitivity-based method [19], the repeated power flow (RPF) method [20], and the direct data-driven method [21]. These methods are summarized in Table 3. M1 is to directly incorporate the DAEs into the optimization problem by adopting the implicit integration rule. M2 uses trajectory sensitivity to achieve TTC calculation. M3 gets the TTC by gradually increasing the generator power base on the initial state and repeatedly calculating the power flow until a certain constraint is about to be violated. And, M5 applies NNs to learn the mapping between system state variables and TTC values. Moreover, to manifest the superiority of our methods, we have advanced experiments under single- and multi-contingency conditions, of which the outcomes are respectively visualized in Figure 4a,b.

Table 3. Different methods and pre-contingencies for TTC calculation.

Methods	TSCOPF	The Sensitivity-Based Method	The Repeated Power Flow Method	TSCOPF with DBN-Assisted
Symbol	M1	M2	M3	M4
	single contingency		multi contingency	
Line	1–39		1–39, 2–3, 3–18, 16–17	

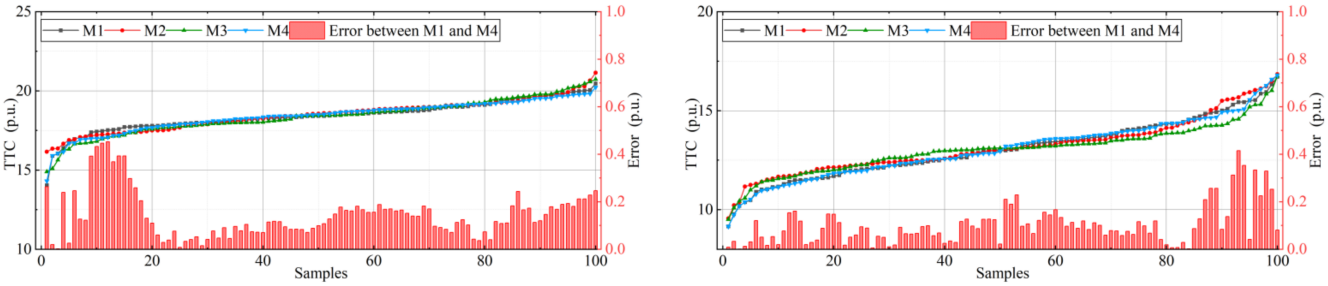


Figure 4. The results of TTC calculation under four different methods: (**a**) Single contingency; (**b**) Multi contingencies.

Figure 4 shows the TTC values calculated by the applied methods under 100 unseen scenarios. The samples are sorted according to the ascending order of the TTC value calculated by M1 to facilitate viewing, and the histogram shows the error between the TTC values calculated by M1 and M4. Taking Figure 4b as an example, the TTC error of M4 is within the acceptable range of [0, 0.5 p.u.], and the average error is 0.1019 p.u. The TTC average errors of M2 and M3 are 0.2308 p.u. and 0.3547 p.u., respectively. Obviously, the TTC value calculated by M4 has the smallest error among several comparison methods. It illustrates that the proposed learning-aided OPF based method can calculate the TTC value more accurately than the RPF and sensitivity-based method. This is because M4, like M1, is modeled based on TSCOPF, which can better describe the system state and has better fidelity than M2 and M3. In addition, it can search the extreme operating point more accurately.

Furthermore, to verify the accuracy of the proposed method, it is compared with the direct data-driven approach (symbol as M5). M5 takes the TTC calculated by M1 as the sample label. Then, it utilizes the DBN model to learn the implicit relationship between the input feature $\mathbf{X}$ and the target feature $\mathbf{Y}_{\mathrm{TTC}}$ and forms a mapping. Figure 5 shows the comparison results of M4 and M5 when the TTC calculated by M1 is used as the reference value. It can be found that M4 has a smaller average relative error, 0.1019 p.u., than M5, which is 0.3297 p.u., in 100 test samples. It means the proposed method has better fidelity than direct data-driven methods.

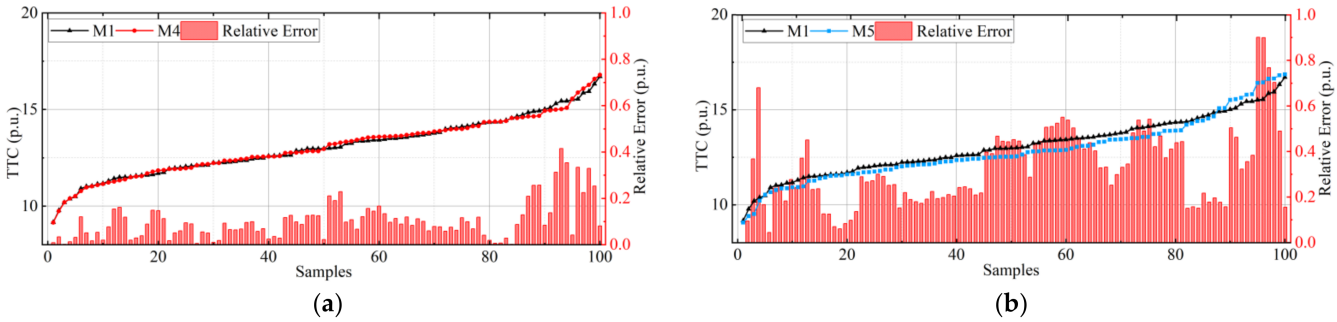


Figure 5. The results of TTC calculation under four different methods: (**a**) M4 compared with M1; (**b**) M5 compared with M1.

In addition, time-domain simulations are performed for each test sample to verify that the operation obtained when calculating the TTC value satisfies the TSCs. The post -fault transient trajectory of rotor angle differences between the individual generators is recorded. The result of a typical sample is shown in Figure 6, where Figure 6a is the transient trajectories after sample initial power flow calculation. Then, utilize the proposed method to calculate the TTC of this sample, and a new operating condition, whose transient trajectories are shown in Figure 6b, can be obtained. It can be observed that the curves have apparent fluctuations. The angle difference between Gen34 (the generator on bus 34) and Gen39 (the generator on bus 39) has the most significant change and is close to the set stability threshold, 180 degrees. It means that the system is operating at its TS boundary at this time. In addition, Figure 6b illustrates that DBN can accurately estimate TSI, and the sensitivity of transient stability margin can help OPF find boundaries of the system. It demonstrates that the learning-aided model can follow the TSCs effectively when it calculates the TTC. Furthermore, the learning-aided model can help the TSCOPF accurately find the most extreme operating condition.

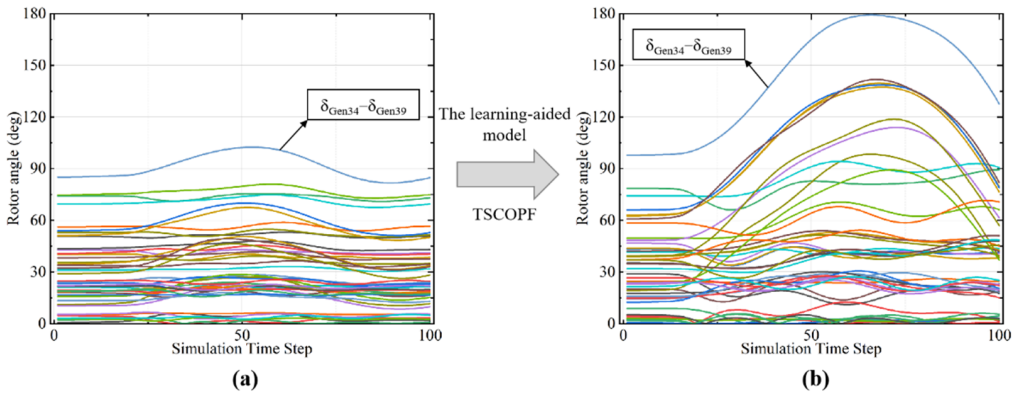


Figure 6. The post-fault transient trajectories of rotor angle differences of the test sample: (**a**) the trajectories after sample initial power flow calculation; (**b**) after TTC calculation.

5.4. Efficiency Comparison

Figure 7 depicts solving time statistics, where the X-axis represents the number of faults in pre-contingency, and the Y-axis is computation time. As shown in Figure 7, the runtime of M1, M2, and M3 are significantly longer than that of M4 under single contingency (i.e., one fault in pre-contingency). And, all algorithms consume more time to compute TTC with more contingencies considered, except for M4. This is because M1, M2, and M3 all need to calculate DAEs associated with TSCs in iterations, and the dimensions of DAEs are higher as more contingencies are considered. However, M4 surrogates the time-consuming part by learning-aided model, and reduces the computation time. The results claim that the proposed method significantly outperforms other comparative methods with respect to efficiency.

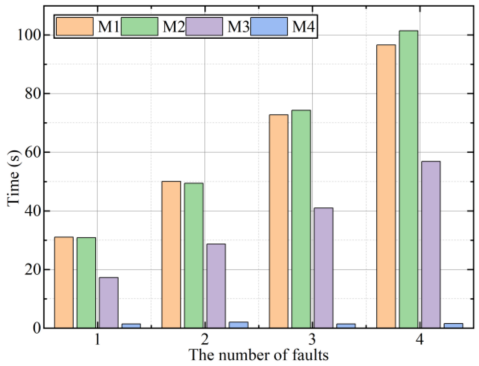


Figure 7. Efficiency analysis.

6. Conclusions

Large-scale wind power penetration has increased the potential insecurity risk of inter-area power exchange. Therefore, rapid and accurate security assessment for inter-corridors is imperative. Towards this end, this paper proposes a learning-aided method for fast TTC calculation. The TTC calculation is firstly modeled as transient stability constrained optimal power flow. Then, to reduce the complexity of the TSCOPF model, DBN-based learning-aided transient stability assessment is introduced to surrogate high-dimensional and time-consuming time-domain constraints. In the end, the Jacobian and Hessian matrix of the trained learning-aided model is derived; thereby, nonlinear programming is allowed to solve the learning-aided TSCOPF model efficiently.

The result of the case study demonstrates that the learning-aided model can achieve TSA with higher accuracy and generalization. Moreover, the learning-aided TSCOPF model proposed in this paper can obtain more accurate TTC values than RPF, sensitivity-based, and direct data-driven methods. This is because the proposed method can both take into account the fidelity and efficiency of physics- and data-driven modeling by combining the learning-aided model with the OPF. And compared with the heuristic search of RPF, the OPF model can search the extreme operating point more accurately. On the other hand, due to the use of the learning-aided model to surrogate the time-consuming TSA, it has higher computational efficiency than other physics-driven methods, which means that it can be applied online after sufficient offline training. Besides, the proposed method is not limited to TTC-oriented research. Because of its high compatibility with other static or dynamic models, it can be extended to other index calculations in the power system that require a large amount of computation but require high efficiency. Other advanced machine learning algorithms will be used to achieve better calculation performance in our future work. And, it would also be meaningful to optimize and control the TTC.

Author Contributions: Conceptualization, G.Q. and J.L.; methodology, G.Q. and J.L.; software, J.L.; validation, J.L.; formal analysis, J.L.; resources, Y.L.; data curation, J.L.; writing—original draft preparation, J.L.; writing—review and editing, G.Q. and Y.L.; visualization, J.L.; supervision, Y.L.; project administration, X.S.; funding acquisition, Y.L. and X.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

According to the chain rule, $\bigtriangledown \boldsymbol{H}_s$ in (31) is:

$$\begin{aligned} \partial\Psi_c(\boldsymbol{X})/\partial \boldsymbol{x}_i &= [\mathrm{d}\Psi_c(\boldsymbol{X})/\mathrm{d}M(D_{l-1}(\ldots(M(D_1(\boldsymbol{X})))\ldots))] \times \ldots \times [\mathrm{d}D_1(\boldsymbol{X})/\mathrm{d}\boldsymbol{X}] \times [\mathrm{d}\boldsymbol{X}/\mathrm{d}\boldsymbol{x}_i] \\ &= \boldsymbol{w}_l\,[\mathrm{d}_{\boldsymbol{x}}M(D_{l-1}(\ldots(M(D_1(\boldsymbol{X})))\ldots)) \times (\boldsymbol{w}_{l-1}\boldsymbol{e}_i)], \end{aligned} \tag{A1}$$

To simplify (A1), the following functions are defined:

$$\boldsymbol{w}_i{}^1 = \boldsymbol{w}^1\boldsymbol{e}_i \tag{A2}$$

$$A_l(\boldsymbol{X}) = \mathrm{d}_{\boldsymbol{x}}M(D_l(\ldots(M(D_1(\boldsymbol{X})))\ldots)) \times \boldsymbol{w}_l \tag{A3}$$

$$\begin{aligned} \textstyle\prod^{l2}{}_{i=l1}A_i(\boldsymbol{X}) &= A_{l1} \times \ldots \times A_{l2}, && \text{if } l_2 < l_1 < l, \\ &= A_{l2} \times \ldots \times A_{l1}, && \text{if } l_1 < l_2 < l, \\ &= 1, && \text{if } l_2 < l \le l_1, \\ &= A_{l1}, && \text{if } l_1 = l_2 \end{aligned} \tag{A4}$$

where $\mathrm{d}_{\boldsymbol{x}}M(D_l(\ldots(\boldsymbol{X})\ldots))$ in (A3) can be derived from the following matrix formulation:

$$\begin{aligned} \mathrm{d}_{\boldsymbol{x}}M(D_l(\ldots(\boldsymbol{X})\ldots)) = \{ &[\mathrm{d}_{Dl\_1(\ldots(\boldsymbol{X})\ldots)}M(D_l{}^1(\ldots(\boldsymbol{X})\ldots)), 0, \ldots, 0]; \\ &[0, \mathrm{d}_{Dl\_2(\ldots(\boldsymbol{X})\ldots)}M(D_l{}^2(\ldots(\boldsymbol{X})\ldots)), \ldots, 0]; \\ &\ldots \\ &[0, 0, \ldots, \mathrm{d}_{Dl\_nl(\ldots(\boldsymbol{X})\ldots)}M(D_l{}^{nl}(\ldots(\boldsymbol{X})\ldots))]\}, \end{aligned} \tag{A5}$$

$$\begin{aligned} D_l(\ldots(\boldsymbol{X})\ldots) &= [D_l{}^1(\ldots(\boldsymbol{X})\ldots); \ldots; D_l{}^{nl}(\ldots(\boldsymbol{X})\ldots)] \in \mathrm{R}^{nl \times 1}, \\ D_l{}^k(\ldots(\boldsymbol{X})\ldots) &= \boldsymbol{w}_l{}^kM(D_{l\text{-}1}(\ldots(\boldsymbol{X})\ldots)) + b_l{}^k \end{aligned} \tag{A6}$$

Using (A2)~(A4), the *Jacobian* matrix can be simplified to:

$$\partial\Gamma_c/\partial \boldsymbol{x}_i = \boldsymbol{w}_l[\textstyle\prod\prod^2{}_{i=l-1}A_i(\boldsymbol{X})] \times [\mathrm{d}_{\boldsymbol{x}}M(D_1(\boldsymbol{X})) \times \boldsymbol{w}_1{}^i], \tag{A7}$$

Similar to the derivation process of the *Jacobian* matrix, the *Hessian* matrix can be obtained by the following formulations:

$$\begin{aligned} \mathrm{d}_{xj}A_l(\boldsymbol{X}) &= \mathrm{d}^2{}_{\boldsymbol{x},xj}M(D_l(\ldots(\boldsymbol{X})\ldots)) \times \boldsymbol{w}_l \\ &= \{ [\mathrm{d}^2{}_{\boldsymbol{x},xj}M(D_l{}^1(\ldots(\boldsymbol{X})\ldots)), 0, \ldots, 0]; \\ &\quad [0, \mathrm{d}^2{}_{\boldsymbol{x},xj}M(D_l{}^2(\ldots(\boldsymbol{X})\ldots)), \ldots, 0]; \\ &\quad \ldots \\ &\quad [0, 0, \ldots, \mathrm{d}^2{}_{\boldsymbol{x},xj}M(D_l{}^{nl}(\ldots(\boldsymbol{X})\ldots))] \} \end{aligned} \tag{A8}$$

$$\begin{aligned} \mathrm{d}^2{}_{\boldsymbol{x},xj}M(D_l{}^k(\ldots(\boldsymbol{X})\ldots)) = [\mathrm{d}^2M(D_l{}^k(\ldots(\boldsymbol{X})\ldots))/\mathrm{d}(D_l{}^k(\ldots(\boldsymbol{X})\ldots))^2] \times \boldsymbol{w}_l \times \\ \textstyle\prod^2{}_{i=l-1}A_i(\boldsymbol{X}) \times [\mathrm{d}_{\boldsymbol{x}}M(D_1(\boldsymbol{X})) \times \boldsymbol{w}_1{}^j], \end{aligned} \tag{A9}$$

where, $\boldsymbol{w}_l = \boldsymbol{w}_1{}^i$ if $l = 1$ in (A8) and (A9). And, defined (A10) as follows:

$$\begin{aligned} \Lambda_k &= [\textstyle\prod^{l-1}{}_{i=k+1}A_i(\boldsymbol{X})] \times \mathrm{d}_{xj}A_k(\boldsymbol{X}) \times [\prod^1{}_{i=k\text{-}1}A_i(\boldsymbol{X})], \text{ if } k \ge 2, \\ &= [\textstyle\prod^{l-1}{}_{i=k+1}A_i(\boldsymbol{X})] \times \mathrm{d}_{xj}A_k(\boldsymbol{X}), \text{ if } k = 1, \\ &= \mathrm{d}_{xj}A_k(\boldsymbol{X}) \times [\textstyle\prod^1{}_{i=k-1}A_i(\boldsymbol{X})], \text{ if } k = l-1 \end{aligned} \tag{A10}$$

Then, the Hessian matrix can be derived as (A11):

$$\partial^2\Gamma_c/(\partial \boldsymbol{x}_i\partial \boldsymbol{x}_j) = \boldsymbol{w}_l[\Sigma^{l-1}{}_{k=1}\Lambda_k] \tag{A11}$$

References

1. Qiu, G.; Liu, Y.; Zhao, J.; Liu, J.; Wang, L.; Liu, T.; Gao, H. Analytic Deep learning-based surrogate model for operational planning with dynamic TTC constraints. *IEEE Trans. Power Syst.* **2020**, *36*, 3507–3519. [CrossRef]
2. Zhang, X.; Santiago, G. Decentralized total transfer capability evaluation using domain decomposition methods. *IEEE Trans. Power Syst.* **2016**, *31*, 3349–3357. [CrossRef]

3. Tang, L.; Sun, W. An automated transient stability constrained optimal power flow based on trajectory sensitivity analysis. *IEEE Trans. Power Syst.* **2017**, *32*, 590–599. [CrossRef]
4. Min, L.; Ali, A. Total transfer capability computation for multi-area power systems. *IEEE Trans. Power Syst.* **2006**, *21*, 1141–1147. [CrossRef]
5. Qiu, G.; Liu, Y.; Liu, J.; Wang, L.; Liu, T.; Gao, H.; Shafqat, J. Surrogate-assisted optimal re-dispatch control for risk-aware regulation of dynamic total transfer capability. *IET Gener. Transm. Distrib.* **2021**, *15*, 1949–1961. [CrossRef]
6. Gan, D.; Robert, J.T.; Ray, D.Z. Stability-constrained optimal power flow. *IEEE Trans. Power Syst.* **2000**, *15*, 535–540. [CrossRef]
7. La Scala, M.; Trovato, M.; Antonelli, C. On-line dynamic preventive control: An algorithm for transient security dispatch. *IEEE Trans. Power Syst.* **1998**, *13*, 601–610. [CrossRef]
8. Yan, X.; Yin, M.; Zhao, Y.; Zhang, R.; David, J.H.; Zhang, Y. Robust dispatch of high wind power-penetrated power systems against transient instability. *IEEE Trans. Power Syst.* **2018**, *33*, 174–186.
9. Yan, X.; Ma, J.; Zhao, Y.; David, J.H. Robust transient stability-constrained optimal power flow with uncertain dynamic loads. *IEEE Trans. Power Syst.* **2017**, *32*, 3415–3426.
10. Alejandro, P.; Claudio, R.F.; Enrique, A.Z.; Jose, M.L. Directional derivative-based transient stability-constrained optimal power flow. *IEEE Trans. Smart Grid.* **2017**, *8*, 1911–1921.
11. Liu, Y.; Zhao, J.; Xu, L.; Liu, T.; Qiu, G.; Liu, J. Online TTC estimation using nonparametric analytics considering wind power integration. *IEEE Trans. Power Syst.* **2019**, *34*, 494–505. [CrossRef]
12. Sun, H.; Zhao, F.; Wang, H.; Wang, K.; Jiang, W.; Guo, Q.; Zhang, B.; Louis, W. Automatic learning of fine operating rules for online power system security control. *IEEE Trans. Neural Netw. Learn. Syst.* **2016**, *27*, 1708–1719. [CrossRef] [PubMed]
13. Zhu, L.; David, J.H.; Lu, C. Hierarchical deep learning machine for power system online transient stability prediction. *IEEE Trans. Power Syst.* **2020**, *35*, 2399–2411. [CrossRef]
14. James, J.Q.Y.; David, J.H.; Albert, Y.S.; Gu, J.; Victor, O.K.L. Intelligent time-adaptive transient stability assessment system. *IEEE Trans. Power Syst.* **2018**, *33*, 1049–1058.
15. Jin, Y. Surrogate-assisted evolutionary computation: Recent advances and future challenges. *ISwarm Evol. Comput.* **2011**, *1*, 61–70. [CrossRef]
16. Jin, Y.; Wang, H.; Tinkle, C.; Guo, D.; Kaisa, M. Data-driven evolutionary optimization: An overview and case studies. *IEEE Trans. Evol. Comput.* **2019**, *23*, 442–458. [CrossRef]
17. Raphael, T.H.; Diana, V.; Anirban, C. Parallel surrogate-assisted global optimization with expensive functions-a survey. *Struct. Multidiscipl. Optim.* **2016**, *54*, 3–13.
18. Cai, X.; Gao, L.; Li, X. Efficient generalized surrogate-assisted evolutionary algorithm for high-dimensional expensive problems. *IEEE Trans. Evol. Comput.* **2020**, *24*, 365–379. [CrossRef]
19. Hamoud, G. Assessment of available transfer capability of transmission systems. *IEEE Trans. Power Syst.* **2000**, *15*, 27–32. [CrossRef]
20. Piyush, W.P.; Sachin, K.; Sinha, A.K. Total transfer capability calculation using modified repeated power flow method. In Proceedings of the 2015 Annual IEEE India Conference (INDICON), New Delhi, India, 17–19 December 2015; IEEE: Piscataway, NJ, USA, 2015.
21. Panagiotis, P.; Theofilos, P.; Andreas, C.; Jovica, M. Measurement based method for online characterization of generator dynamic behaviour in systems with renewable generation. *IEEE Trans. Power Syst.* **2018**, *33*, 6466–6475.
22. Nattawut, P.; Akihiko, Y.; Yoshiki, N.; Verma, S.C. Improved risk-based TTC evaluation with system case partitioning. *Int. J. Electr. Power Energy Syst.* **2013**, *44*, 530–539.
23. Lukmanul, H.; Junji, K.; Yue, Y.; Tomohisa, M.; Yoshifumi, Z.; Naoto, Y.; Yoshihito, N.; Kimihiko, S.; Akira, T. A study on the effect of generation shedding to total transfer capability by means of transient stability constrained optimal power flow. *IEEE Trans. Power Syst.* **2009**, *24*, 347–355.
24. Pandia, J.; Chanan, S. Reliability constrained multi-area adequacy planning using stochastic programming with sample-average approximations. *IEEE Trans. Power Syst.* **2008**, *23*, 504–513.
25. Zhang, C.; Pin, L.; Qin, A.K.; Kay, C.T. Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics. *IEEE Trans. Neural Netw. Learn. Syst.* **2017**, *28*, 2306–2318. [CrossRef] [PubMed]
26. Jaber, A.; Yushi, M.; Toshifumi, I. Assessment and optimization methods for microgrid with multiple VSG units. *IEEE Trans. Smart Grid.* **2018**, *9*, 1462–1471.
27. Huang, H.; Chung, C.Y. Coordinated damping control design for DFIG-based wind generation considering power output variation. *IEEE Trans. Power Syst.* **2012**, *27*, 1916–1925. [CrossRef]
28. Antonio, L.B.; Glauco, N.T.; Djalma, M.F. Simultaneous tuning of power system damping controllers using genetic algorithms. *IEEE Trans. Power Syst.* **2000**, *15*, 163–169.
29. Andrei, P.; Iman, S.; Chris, M. Interior Point Differential Dynamic Programming. *IEEE Trans. Control Syst. Tech.* **2021**, *29*, 2720–2727.