WindGISKI: Using AI to Propose Areas Suitable for Building New Wind Turbines

Daniel Gritzner*, Sandra Peters-Erjawetz‡, Carsten Fichter‡, and Jörn Ostermann*
*Institut für Informationsverarbeitung, Leibniz Universität Hannover, Germany
{gritzner,ostermann}@tnt.uni-hannover.de
‡Hochschule Bremerhaven, Germany
{speters,cfichter}@hs-bremerhaven.de

***Abstract*—To mitigate the effects of man-made climate change a switch to renewable energy sources is necessary. However, new wind turbines are not built as quickly as is necessary in Germany. There are many sources of delays or even conflicts, e.g., slow approval processes, lack of public acceptance, or wildlife conservation laws. In the project WindGISKI we therefore developed an AI model which is able to predict wind farm suitability scores for areas in Germany, as well as which features are most important for the model's prediction. By feeding this information into a geographic information system (GIS), this system can assist end users, such as region planners in local authorities or employees of wind energy companies, in their decision making and thus speed up the transition to renewable energy. In this paper we present a survey we conducted among industry experts, our AI model and a work-in-progress prototype of our vision of an AI-enhanced GIS. The expert survey was necessary to identify suitable samples for training our AI. It showed that urban structure and nature preservation are most relevant to wind energy projects, while social factors are barely relevant. Additionally, we designed a new metric for measuring our model's performance in the light of a very drastic class imbalance of samples which rendered existing metric unsuitable.**

***Index Terms*—artificial intelligence, deep learning, geographic information system, renewable energy, wind energy**

I. Introduction

The German government aims for the country to become neutral in terms of greenhouse gas emissions by 2045 in order to minimize the effects of man-made climate change. Producing energy from renewable sources, such as wind and solar energy, is an essential aspect in the strategy to achieve emission neutrality. However, the actual amount of new wind turbines being built every year lags behind the target metrics set by the government. Multiple reasons prevent a faster expansion of wind energy in Germany, e.g., slow approval of permissions for wind farm projects, high construction costs, a lack of public acceptance of wind turbines, or conflicts with nature protection laws or laws regulating disturbances caused by exposure to noise or shadows [1], [2]. As a consequence, too few potential areas for wind farms are proposed by local authorities and the actual wind energy projects are often delayed beyond their initial target date due to lawsuits. Even though the wind energy industry gained a lot of experience in the recent decades, the average time from starting the pre-planning of a new wind farm to finishing its construction increased from 5.5 years in 2015 to 8 years in 2022 [3].

This research is funded by the Bundesministerium für Umwelt, Naturschutz, nukleare Sicherheit und Verbraucherschutz (BMUV).

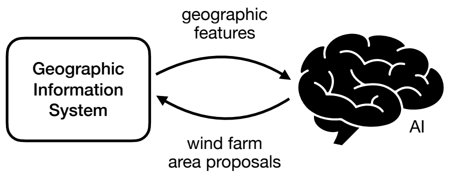

Fig. 1. The project WindGISKI is about using an AI to enhance a GIS to assist in finding new, suitable areas for wind farms.

The project WindGISKI[1] aims to use artificial intelligence (AI) to enrich a geographic information system (GIS) with additional information in order to assist people involved with wind energy projects, e.g., employees at local authorities or at wind energy companies, in their decision making process, as shown in Fig. 1. The goal is to identify low conflict areas in Germany on which new wind farms can be built quickly.

The interdisciplinary project team consists of research institutes from several fields, including engineering, computer science, social sciences, and life sciences, as well as companies working in the renewable energy business. Due to the broad range of interests affecting wind energy, a wide range of knowledge is necessary for the success of WindGISKI. The project covers many aspects: identifying features relevant to wind energy projects, collecting data suitable for AI model development, conducting interviews and surveys with industry experts for validation, implement noise propagation simulations, and more. In the end, the goal is not only to have an expert-validated AI model proposing areas suitable for wind farms, but to also have a separate reference booklet which instructs users of the AI on how to use it, as well as what other factors to consider when realizing a wind energy project that an AI cannot cover. As an example, involving the local population in a project such that it directly benefits from the nearby wind farm increases the acceptance and therefore how quickly the turbines can be built.

This paper specifically covers the AI model we developed for WindGISKI, our vision of how users may interact with the AI, and what information the AI can provide to assist

[1]https://www.windgiski.uni-hannover.de/

decision making. The remaining structure is as follows: in the next section we will discuss related work. Then we describe our method, i.e., the actual AI model and surrounding aspects relevant to its development. In the fourth section we will evaluate our model before finishing with a conclusion.

II. Related Work

The combination of an AI and a GIS has already been proposed in many cases. The closest work to ours is a very similar work using an AI-enhanced GIS to assist wind farm planners in Tuscany, Italy [4]. However, their spatial resolution is far coarser than ours at $10km$ compared to our $50m$. The next closest is a work predicting suitable wind turbine locations in the USA [5]. The authors are mainly concerned with studying the relationship of wind energy and wildlife conservation, though. Another work is concerned with predicting wind energy potential using AI and geospatial data [6]. Wind energy potential is an input feature for our model. Other works combining AI and GIS include efforts to improve decision making in other domains, e.g., city planning or disaster management [7]–[11]. A survey of state-of-the-art uses of geospatial data, including AI-based ones, can be found in [12]. A novel approach of combining AI with a GIS is using a large language model as a user interface to a GIS to help end user with creating visualizations or evaluations using existing data within a GIS [13].

III. Method

In this section we first discuss our dataset and the difficulties we faced with collecting suitable data for training an AI model. A survey conducted among industry experts, which we present in the second subsection, helped mitigate some of these difficulties. In the third subsection we explain the models we evaluated and in the last subsection we present our vision of how end users should be able to interact with our AI.

A. Dataset

In an initial phase of WindGISKI the project partners collected a list of features relevant to the success of a wind farm construction project. Each feature was classified as relevant to the AI model and/or relevant to the reference booklet, which was mentioned in the introduction, based on whether the feature is actually measurable and if so, measurable at a large scale, i.e., for the entirety of Germany. Features representing clear proposals without any room for decisions, such as involving the local population and having it directly benefit from the new wind turbines, were classified as only relevant to the reference booklet. Features from the categories meteorology, bodies of water, landscape preservation, nature preservation, wildlife conservation (birds and bats), forests, structure of urban development, traffic infrastructure, power grid infrastructure, topography, aviation, and military concerns were used by the AI model. Some features were discrete classifications, e.g., is an area a legally designated nature preservation area, while other features were continuous, e.g., distance to the closest residential building. A small subset of features was omitted entirely due to funding guidelines: WindGISKI must be non-political. Therefore features such as the voting behavior of the local population were omitted.

All of the data relevant to the AI was collected in a geo-referenced form, mostly as polygons. In order to be able train a model on this data we partitioned Germany in cells of size $50m \times 50m$ resulting in a tensor of shape $[H, W, C]$ with the height $H = 17359$, width $W = 12818$, and the feature dimension C. We stored each feature as a separate image in order to be easily able to choose which feature to load at each step of the AI training pipeline and which feature to omit to save processing time and memory. We converted all polygon coordinates to the EPSG:4839 coordinate system and rasterized all features into the data tensor. Data which was already rasterized was reprojected to the same coordinate system and resolution as our data tenstor.

We used a subset of $|C_e| = 33$ features to determine cells on which no wind turbine can be built for any reason, e.g., economic reasons (insufficient wind speed) or legal reasons (presence of residential buildings). For the actual AI model we used $|C_m| = 57$ features with some overlap between the features in C_e and C_m. Again, residential buildings are an example: they prevent wind turbines from being built in the same location (C_e) but the distance to the closest such building is also relevant to how suitable a cell is for wind turbines (C_m).

A naive approach for training a model would be a binary classification of cells based on whether they are part of an existing wind farm (positive class) or not (negative class). However, this approach is unsuitable since the premise of the project is that cells exists which are suitable for wind farms but no turbines have been built there yet. Therefore, we needed a way to identify samples which actually represent the negative class well. The cells omitted due to C_e are unsuitable as negative samples since the model would at best learn to look for overlapping features in C_e and C_m and therefore learn to identify what we already know. To mitigate this problem, we used the results of a survey we conducted among industry experts to create a rough scoring of cells to identify those cells which are likely negative samples but which are not excluded due to the presence of a feature in C_e.

We used wind farm cells as positive samples. However, we face a similar problem as we do with the negative samples. Due to advances in technology and changes in Germany's legal framework some existing wind turbines would no longer be built nowadays. We created a filtered subset of all existing wind turbines in which we excluded all wind turbines which were commissioned before January 1st, 2010, or whose total height (hub height plus rotor radius) is less than $150m$. The age filter accounts for the change in the legal framework. The height filter was used to exclude wind turbines which can no longer be built and run in an economically feasible way today.

Our dataset only contains coordinates of wind turbines but no wind farm identifiers as that concept is usually not represented in databases such as the Marktstammdatenregister. We therefore used the following approach based on heuristics provided by industry experts. We placed an ellipse around each wind turbine with its major axis aligned with the prevalent wind direction. For simplicity and based on expert knowledge, we assumed south-west winds as prevalent wind direction for all of Germany. The radius along the major axis was $5D$ and $3D$ along the minor axis with D being the rotor diameter. This ellipse represents the area in which no other wind turbine should be built due to negative interactions between nearby wind turbines. However, since wind farms are built compactly to maximize the number of turbines in a farm, these ellipses will overlap for close turbines within the same wind farm. Therefore, we use the overlapping of these ellipses as criterion to decide whether two turbines belong to the same wind farm.

Due to our assumptions wind farm assignments can be efficiently computed by rotating our coordinate system such that the first axis aligns with the prevalent wind direction and the second axis aligns with the minor axis. By scaling all coordinates by $\frac{1}{5}$ and $\frac{1}{3}$ along the respective axes all ellipses become circles of radius D in this transformed space. As a result, we only need to compare the Euclidean distance of any two turbines in this transformed space to the sum of their rotor diameters to decide whether their ellipses overlap or not.

After identifying wind farms, we computed the convex hull of each wind farm and intersected this hull with the union of all the wind turbine ellipses. The resulting area was used as the wind farm area. The intersection of the convex hull and the ellipses was necessary to get a better approximation of some wind farms, e.g., those shaped roughly like the letter L. We then rasterized the resulting wind farm areas into our data tensor. We performed this entire process once without either of the previously described age and height filters and once with both filters active. The difference in area between the two variants was marked as "ignored". This area contains a wind farm so we do not want to use it for negative samples but due to the aforementioned reasons we do not want to use it for positive samples either.

To compute distances to a feature, e.g., the distance to residential buildings or forests, we used an algorithm based on parabola intersections to compute a distance field for the respective feature [14], [15].

B. Expert Survey

To validate the results of WindGISKI's AI, interviews with and a survey among experts were conducted. The interviews ensured that no relevant features were missing on our list. We then conducted an online survey among industry experts. The link to the survey was sent to almost 900 persons, 66 of which responded. 45 of those 66 persons answered the survey fully, while the rest chose to drop out at some point. Due to the low number of full responses we consider our survey to be not be representative but still show trends. The respondents were employees or members of a diverse set of groups, e.g., wind energy companies, local authorities, nature conversation organizations, or law firms. According to the self-reported experience, about 50% of respondents have more than ten years of experience in wind energy and about 23% of respondents have between six and ten years of experience. Participants were asked about their goals in the wind energy industry, for whom they think our AI may be useful, how relevant each feature category is, and to rate each feature within each category. The last two questions, i.e., feature category relevance and feature rating, are particularly important for our AI model training.

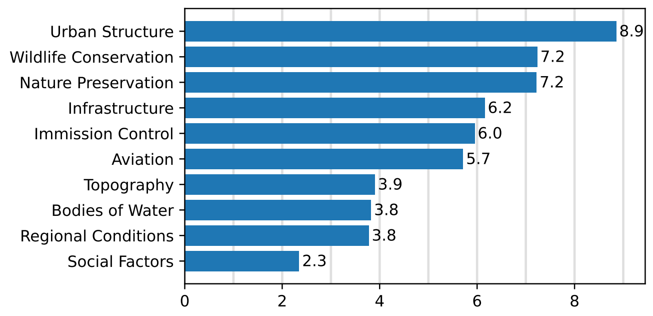

Fig. 2. Mean relevance of feature categories according to a survey we conducted among experts. We omitted the variance since we do not consider the survey representative.

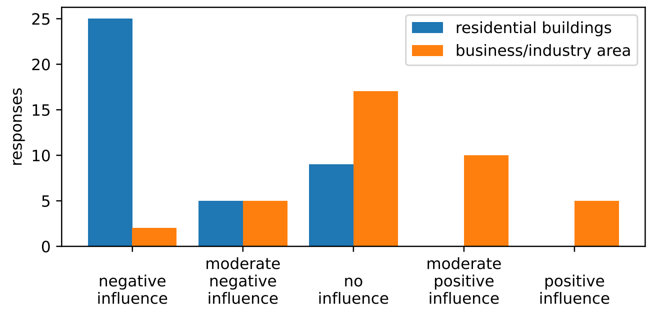

Fig. 3. Histogram of expert ratings of two features from the category "urban structure". The close proximity of residential buildings is considered to affect wind farm projects negatively while the close proximity to businesses or industry on average has no influence according to experts. There is a slight bias towards a positive influence, though.

For the feature relevance participants were asked to assign an importance or relevance with regard to their goals to each category on a ten step Likert scale from 1 (low relevance) to 10 (high relevance). The results are shown in Fig. 2. The most relevant categories are urban structure (where people live and work), wildlife conservation (e.g., birds and bats), and nature preservation. Social factors such as age distribution of the nearby population are of relatively low concern.

For the feature rating participants were asked to rate each feature on a five step Likert scale with regard to their goals:

1) Negative influence
2) Moderate negative influence
3) No influence
4) Moderate positive influence
5) Positive influence

Each feature was assigned to exactly one of the categories in Fig. 2 and all features were grouped by category in the survey. We also included all features we did not deem relevant to the AI in our dataset. A small excerpt of the results is shown in Fig. 3. The responses showed that some features have a negative influence, e.g., proximity to residential buildings (urban structure), while others affect wind energy projects positively, e.g., trust in local authorities (social factors). This even translated into trends for entire categories. Certain categories like urban structure or aviation largely contained features with a negative influence while social factors were deemed to mostly have a positive influence.

As mentioned in the previous subsection, choosing reliable negative samples for AI model training from our dataset was an issue. We therefore used the expert survey results to implement a rough scoring of each cell from our dataset to identify negative samples. In our scoring model, we assigned a factor ω_c to each category and to each feature ω_f. The score assigned to a cell is

$$\sum_{f \in F} \omega_{c(f)} \cdot \omega_f$$

with the subset F of features relevant to that cell and $c(f)$ being the category f is in. The distance up to which each feature is considered relevant, e.g., up to which distance a residential building is considered to be close and therefore relevant, was determined by an industry expert.

To determine the factors ω_c and ω_f we used the hyperparameter optimization tool SMAC [16]. First, we computed the relative amount of responses for each feature category c and each response $i \in \{0, 1, 2, \dots, 9\}$ such that c_i is the relative amount of responses that rated the category c at $10 - i$. As an example, 63 out of 66 survey participants answered the question regarding the category $c =$ "urban structure". 41 out of those 63 responses rated the relevance of this category as 10, therefore $c_0 = \frac{41}{63} \approx 0.65$. We also determined relative amounts of responses f_i for each feature in the same way.

We modelled ω_c has having an exponential decay in relevance, i.e.,

$$\omega_c = \sum_{i=0}^{9} c_i \cdot e^{-\lambda \cdot i}$$

with the hyperparameter λ being one of two hyperparameters optimized by SMAC. We modelled ω_f as

$$\omega_f = -\alpha \cdot f_1 - \beta \cdot f_2 + \beta \cdot f_4 + \alpha \cdot f_5$$

with $\alpha = 0.5 + \gamma$ and $\beta = 0.5 - \gamma$ where $\gamma \in (0, 0.5)$ is the second hyperparameters opitmized by SMAC. In this equation f_1 is the relative amount of "negative influence" responses, f_2 is the relative amount of "moderate negative influence", and so on (cf. the enumeration earlier in this subsection).

This modelling was chosen such that an emphasis is put on the higher ratings wrt. to ω_c and such that α and β are values between 0 and 1 with $\alpha > \beta$. Furthermore, ratings of negative influence (f_1 and f_2) lead to negative scores, ratings of no influence (f_3) were ignored, and ratings of positive influence (f_4 and f_5) lead to positive scores.

SMAC needs an optimization goal to be able to optimize hyperparameters. As a maximization goal, we choose the relative number of cells being assigned a score less than the mean score of the non-ignored wind farm cells. The hyperparameter values $\lambda \approx 0.359$ and $\gamma \approx 0.023$ maximize this goal with about 82.8% of cells being assigned a score less than the non-ignored wind park mean score. With these hyperparameter settings we computed a rough scoring of all cells in our dataset which we then used later to choose reliable negative samples for our AI model training.

C. AI Model

The goal of WindGISKI is to train an AI which proposes suitable areas for constructing wind farms. However, we do not simply want to replicate decisions by experts but rather to use the rough scoring from the previous subsection as a guide. The goal is for the AI to be able to discover new suitable areas which experts have not discovered yet so far. Therefore we implemented three different AI model variants with different levels of reliance on the survey-based scoring.

The first two variants are based on how the model is trained while being flexible wrt. the model architecture. The third variants requires a specific kind of model architecture in addition to a specific way of training:

1) Binary classification
2) Metric learning
3) Normalizing flows

Our first variant treats the problem as a binary classification. The model is trained to assign high logits (= scores) to positive samples taken from non-ignored wind farms and low, even negative, logits to negative samples chosen based on the rough scoring from the previous subsection. We train the model to minimize the cross-entropy

$$H() = - \sum_{cls \in \{neg, pos\}} \mathbb{1}_{y=cls} \cdot \log\left(p(cls|x)\right)$$

where x is a vector of length $|C_m|$ from our data tensor, i.e., a vector describing all the features of a single cell, and $y \in \{neg, pos\}$ defines whether the sample x is a positive or negative sample. $\log\left(p(cls|x)\right)$ is the model's prediction. The logits $\log\left(p(pos|x)\right)$ predicted by a model trained this way can be used as a score for each cell.

To choose which samples to use as negative samples we first determined the number of positive sample cells n in each federal state. Since federal state laws differ wrt. to wind energy we chose as many negative samples from each federal state as there positive samples in that state. We ordered all candidate cells, i.e., cells not excluded (cf. C_e in subsection III-A) which are not part of an existing wind farm, by their expert-based score in ascending order. We then randomly chose n negative samples from the bottom $3n$ samples of this ordered list, i.e., we randomly chose from a subset of candidates likely to be good negative samples.

Modern models in other domains such as natural language processing or computer vision often are based on a transformer architecture such as OpenAI's famous GPT-line of models [17], [18]. However, these models require a far larger amount of training samples than we can provide. We therefore used

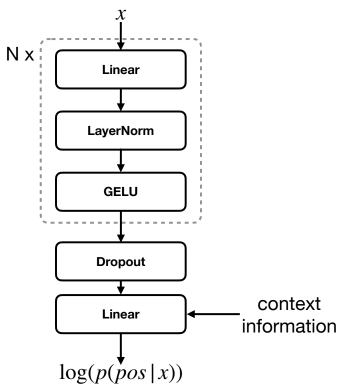

Fig. 4. The architecture of the MLP we evaluated. Details on N (how many times we repeated the Linear-LayerNormalization-GELU-Activation-block) and the hidden dimensions after each Linear layer can be found in section IV. The context information is an optional vector which is concatenated with the output vector of the Dropout layer to form the input of the last Linear layer.

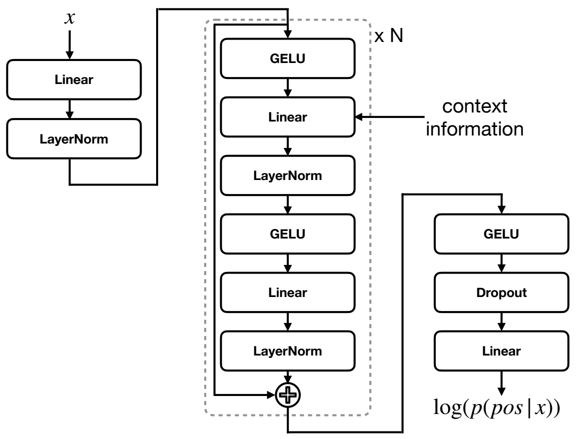

Fig. 5. Another MLP architecture we evaluated. Compared to Fig. 4, this architecture uses residual connections and a larger block that is repeated several times. Again, details on the configuration can be found in section IV. Also, the context information is optional and, if present, integrated in the same way as in the standard MLP, i.e., by concatenation.

an older but still performant multi-layer perception (MLP) [19] architecture as shown in Fig. 4. MLPs are even still used as components within transformers, e.g., the feed forward blocks in [17]. Inspired by the design of modern convolutional neural networks (CNNs) [20], [21], we used layer normalization instead of batch normalization and GELU instead of ReLU as activation function. Therefore, our modernized MLP consists of several blocks, each of which consists of a linear transformation layer followed by a layer normalization layer and a GELU layer. After the final such block, we apply dropout [22] before applying a final linear transformation of the model features into $\log(p(pos|x))$ (it is not necessary to compute $\log(p(neg|x))$ explicitly to compute the binary cross entropy). The input of our MLP is a vector x describing a single cell of our dataset. Optionally, we provide context information describing the surroundings if the cell represented by x. If we do so, we concatenate the context feature vector with the output of the dropout layer before applying the final linear transformation. Since this changes the number of weights of the last layer, in each experiment we decide whether to always use context information or to never use context information.

In order to be able to train deeper models with more overall layers, modern models use residual connections, i.e., the input of certain layers is added to the output of later layers [17], [23]. To take advantage of this, we also evaluated an MLP with residual connections as shown in Fig. 5. Keeping in line with [23] we place the residual connections such that their end, where two signals are added together, are placed just after a normalization and before an activation function application. For the residual connection to be well-defined, the number of features going into the repeated residual block and the number of features going out of the block have to be equal. In order to allow a different number of features, e.g., for applying non-linearities in a higher dimensional intermediate space, each repeated block contains two linear transformations. The first linear layer may map the features to a higher dimensional space, while the second layer then projects the features back into the same space that was used as an input to the block. The rest of the residual MLP design is the same as in our regular MLP (Fig. 4) with the exception of the context information, which, if available, is concatenated to the input of the first linear layer in every residual block.

For the optional context information we start with a small image centered around the cell x for which we want to make a prediction. Each "pixel" of this image is a cell, i.e., this image has $|C_m|$ channels. We use an image classification model as feature extractor, as is common for many computer vision tasks such as segmentation or object detection. We change the first convolutional layer of the classification model to accept $|C_m|$ input features and remove the final classification layers. We flatten the extracted feature map into a vector which is then used as the context information for the MLPs. Again, due to the lack of training data, we use an older, smaller, parameter-efficient model, namely Xception [24], as our feature extractor.

The second training approach we used for our models is based on metric learning. In metric learning a model is trained to directly assign scores to input samples, e.g., siamese networks learn to compute a similarity score for pairs of inputs. In our case, we compute scores for individual cells. We use the exact same network architectures as before, but we use a different loss function. The loss function

$$\mathcal{L}(x_1, x_2) = \mathrm{ReLU}(m(x_2) - m(x_1) + \Delta)$$

compares two samples x_1 and x_2. It penalizes the model m if it assigns a higher score $m(x_i)$ to x_2 than it does to x_1. The score $m(x_1)$ is supposed to be at least Δ larger than $m(x_2)$. Therefore, x_1 is supposed to be a better sample than x_2. We use two kinds of pairs of samples for our loss. First, we use inter-class pairs, i.e., x_1 is a positive sample and x_2 is a negative sample. In this case we use a large value for Δ. Secondly, we use intra-class pairs, i.e., both x_i are from the same class (positive or negative) but x_1 has a higher score than x_2 according to the expert-based rough scoring. We do use a

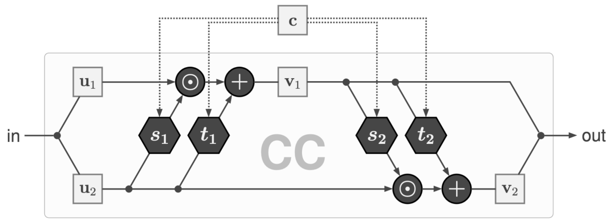

Fig. 6. Coupling blocks which can be used to implement a normalizing flow. Diagram taken from [25].

smaller value for Δ in this case but we do not use the actual difference of the rough scoring. We use the rough scoring as a ranking rather than a true absolute scoring.

The third and last training approach is based on normalizing flows [25]. Normalizing flows are invertible models, i.e., they map vectors of a specific length to vectors of the same length. However, the output vector is actually in the space of a known probability distribution, usually a multivariate standard normal distribution. This can be achieved by a sequence of coupling blocks as shown in Fig. 6. Each coupling block takes an input vector and splits it into two components u_1 and u_2. First, u_2, and optionally context information c, is used to compute the scaling and translation parameters of an affine transformation via the submodules s_1 and t_1. This affine transformation then transforms u_1 into v_1 which is used in a similar fashion to compute an affine transformation of u_2 into v_2. Then, v_1 and v_2 are concatenated into the output of the coupling block. This sequence of affine transformations is reversible.

Assuming that the output of the final coupling block is in the space of a multivariate standard normal distribution, the likelihood of the output vector can be computed. Minimizing the loss function

$$\mathcal{L}(x) = \mathbb{E}\left[\frac{\|m(x)\|_2^2}{2} - \log|J|\right]$$

with the input vector x representing a single cell of our dataset, the normalizing flow model m and the determinant $|J|$ of the Jacobian matrix $\frac{\delta m}{\delta x}$, results in maximizing the likelihood of all samples x shown to the model m. Normalizing flows learn the distribution of all the samples shown to it. The model will learn to assign high likelihoods to wind farm cells and low likelihoods to every cell that is dissimilar. We therefore do not need the rough scoring in this approach at all.

To increase the transparency of the black box AI models we use, we use integrated gradients to compute the importance of each feature. First, we compute the average features of all cells in our dataset as a baseline $\bar{x}$. We then, for each cell, linearly interpolate in multiple steps from the baseline to the actual cell features, compute the loss and backpropagate the gradients to the input vector x. The integrated gradient then is the sum of these input gradients over all interpolation steps. The relative absolute values of the individual features in the integrated gradient measure the relative importance of the features to each other. We further improved this measurement by using SmoothGrad. SmoothGrad applies the integrated gradient computation multiple times to each cell x but adds a small amount of random noise to x each time. The final result is the average of all integrated gradients for a given cell x.

D. Geographic Information System

In Fig. 7 we show a work-in-progress prototype of our vision on how to integrate our AI model's predictions into a GIS. The screenshot at the top shows an excerpt of Germany (northern tip of Germany including the island Sylt) and a user control panel (left-hand side). The excerpt shows Germany as a heatmap (from purple for low scores over blue and green to yellow for high scores) with excluded areas (cf. C_e in subsection III-A) shown in red. Gray is used for cells outside of Germany, including the sea. A zoomed out preview of all of Germany can be seen in the bottom-left of the screenshot.

From top to bottom, the control panel allows the user to select which AI model to use. In the screenshot a model trained solely on data from the federal state Schleswig-Holstein is selected in the drop-down menu. Next, the user is able to adjust the scoring ("Bewertung") used for the heatmap. Options are "absolute" (the score range is mapped to the interval $[0, 1]$ with 0 being rendered in purple and 1 being rendered in yellow), "relative" (each cell x is mapped to a value in $v \in [0, 1]$; v is the relative amount if cells in the training data which has an equal or lower score than the cell x), and "relative to wind farms" (same as "relative" but only the wind farm cells from the training data are used as a reference). The heatmap can even be turned off, rendering all heatmap pixels as black instead. The controls also allow the user to scale certain features (distance to residential buildings is shown in the screenshot) before the model assigns a score to each cell. This allows the user to simulate changes in laws, e.g., if a change in law required all wind turbines to be twice as far away from residential buildings, settings the corresponding feature's factor to 0.5 would simulate this change.

Next, the user is able to choose a target area ("Zielfläche") for further evaluation. The current target area is shown as a magenta square near the center-right of the screenshot. An actual end-user application would require the user to be able to specify areas freely as arbitrary polygons. The evaluation ("Auswerten") button opens a new window, a mock-up of which is shown in the bottom-left of Fig. 7. This window shows the distribution of scores in the target area, as well as which features were most important in the model's decision. A history of recently viewed target areas is also shown for an easy comparison of areas.

The optimization ("Optimieren") button leads to a series of dialogues in which the user can set parameters for areas to propose. Example parameters are the desired shapes of wind farms, the number of wind farms, constraints such as distances to existing wind farms, or the minimum area which should be proposed for new wind farms. An evolutionary algorithm is then used to find a configuration of areas satisfying all constraints and optimizing a desired goal, e.g., maximizing the average score of the proposed cells. As a last step, the user can inspect all areas proposed by the evolutionary algorithm

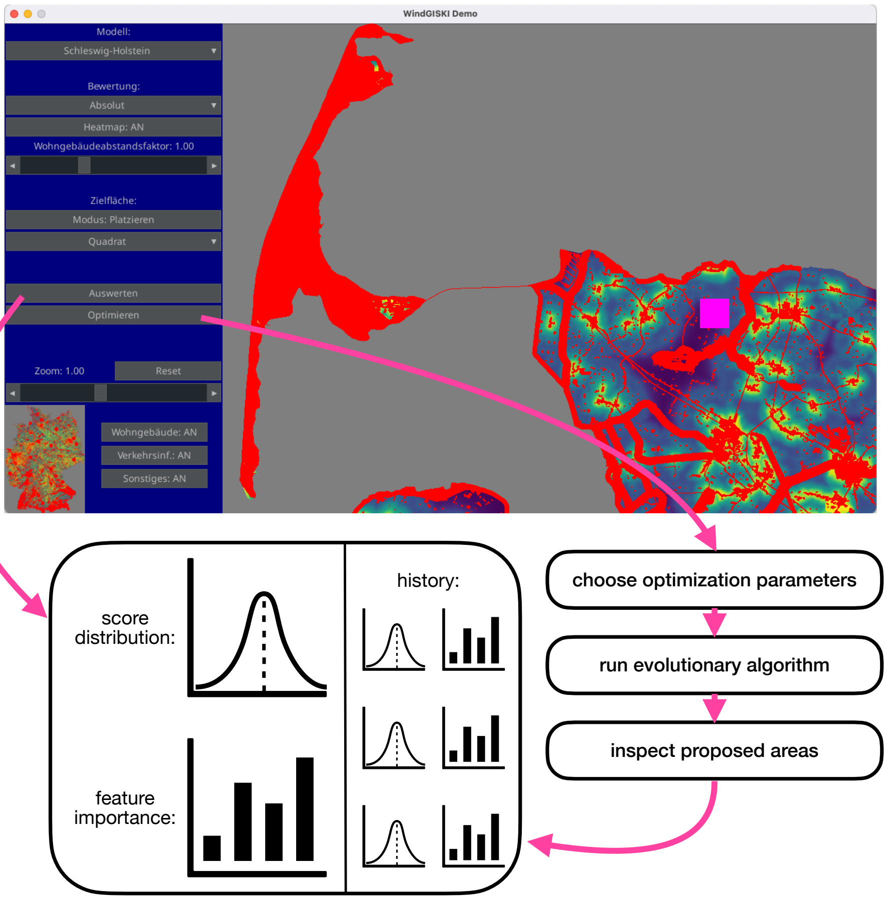

Fig. 7. A work-in-progress prototype demonstrating the integration of our AI model into a GIS. The heatmap uses mock data.

as they could do by choosing a target area directly and using the evaluation button.

Lastly, the last group of controls allows the user zoom the excerpt of Germany. The excerpt can be panned by holding a mouse button and moving the mouse or by clicking on the desired location in the small preview in the bottom-left. Next to the preview are controls allowing the user to disable the exclusion of cells due to certain features in C_e, i.e., fewer cells will be red. Again, this enables more freedom of choice for the user and allows adaptation to changes in laws.

IV. Evaluation

In this section we evaluate our model variants and further inspect the performance of the best variant in the different federal states of Germany. But first we introduce our evaluation metric, since existing metrics only provide a poor signal for optimization and/or comparison of models.

A. Metrics

In each experiment we applied the standard practice of partitioning our training data into an actual training subset and a validation subset (roughly 15% of samples). For the positive samples we assigned each wind farm to either the training subset or the validation subset so that all cells of a wind farm are either used for training or used for validation.

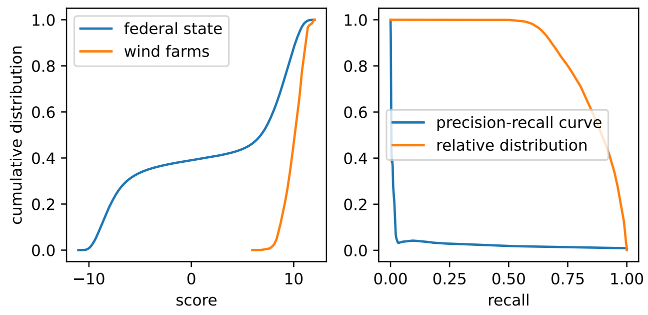

Fig. 8. Distribution of scores of model's prediction (left). The blue curve shows the distribution of all non-excluded cells of an entire federal state while the orange curve only shows the score distribution of the wind farm cells. The right shows two attempts at quantifying the quality of the model. The blue curve is a precision-recall curve used to compute the average precision while the orange curve is used to compute our evaluation metric.

We quickly noticed that regular metrics such as accuracy, precision, or recall did not provide useful information for comparing models and therefore for optimizing hyperparameters such as the number of layers. Almost all models quickly achieved 100% accuracy, even on the validation subset. We tried to shift to the average precision, which is commonly used in object detection. In Fig. 8 an example is shown. To compute the average precision, a precision-recall curve is created by systematically choosing a score threshold to classify cells into positive or negative. The average precision is the area under the precision-recall curve and is a value between 0 (bad) and 1 (good). However, since there are far more non-wind farm cells than wind farm cells in every federal state, the precision quickly drops to very low values. This is to be expected: even in the federal state Schleswig-Holstein, which has a relatively dense distribution of wind farms, there are almost 90 times as many non-excluded non-wind farm cells as wind farm cells. If we assume that just 1% if those cells are actually very suitable for new wind farms, the number of non-wind farm cells in the precision computation quickly outnumbers the total number of wind farm cells in the entire federal state. This can be observed in the example in Fig. 8. However, the actual distribution of scores in the example is actually desirable. The wind farms are assigned high scores while the federal state overall has some good, some mediocre, and some unsuited cells, just as we should reasonably expect.

Our metric uses a similar approach to the average precision to compare the relative location of two probability distributions while being independent of the absolute number of cells/samples in either distribution. We want most of the mass of our distribution of positive samples to be on the higher end of the overall distribution of all samples/cells, just as shown in the left subplot of Fig. 8. For the horizontal coordinate x of the orange curve in the right subplot, we use an approach similar to the recall. We choose score thresholds t such that $x\%$ of all cells in a federal state have a score of t or less. For the vertical coordinate, we compute the relative amount of wind farm cells which have a score equal to or higher than t. We can use the orange curve in the left subplot to do so. The meaning of the orange curve in the right subplot can be interpreted as follows: as you move along the horizontal axis, you go from the worst scores in the federal state to the best scores. The vertical axis then tells you the relative amount wind farms which are at least as good as this score. The curve for our metric is always monotonically decreasing. As the final evaluation metric we compute the area under this curve, just as is done for the average precision. We call our curve the relative distribution and therefore our evaluation metric the average relative distribution (ARD).

B. Model Variants

In a first experiment we compared the three different model training variants with the results shown in Figures 9 and 10. In these experiments we not only randomly picked a model training variant and model architecture, but we also randomly chose the learning rate as a first step in a hyperparameter optimization process. All MLP models were trained with $N = 4$ blocks with a decreasing number of output features of the linear layers $(256, 192, 128, 64)$. The residual MLP model used $N = 8$ blocks with the residual feature dimension set to 128, i.e., the first linear layer in each residual block expected an 128 dimensional input vector (before concatenation of the optional context information if present). The feature dimension between the two linear layers in each residual block was set to 256. The dropout probability was set to 0.1 for both model architectures. For the normalizing flow models, we used ten coupling blocks and all submodules s_i and t_i consisted of two linear layers with a leaky ReLU activation in between and no normalization. The number of input and output features of each submodule was defined by how the input vectors of each coupling block got split into u_1 and u_2. We used 256 features as an intermediate feature dimension between the two linear layers in each submodule. When using optional context information, we used Xception to generate feature maps. Xception eventually increases the feature dimension of the feature maps it computes to 728 and more. We decided to limit all convolutional layers to no more than 256 features to reduce the model size and account for the limited amount of available training data. The “images” created extended 32 cells in each direction (north, east, south, west) from the center cell x which was to be scored. Xception uses strided convolutionas in five places to downsample the input image, i.e., the resulting feature map had a height and width of 2 spatial units (downsampled by $2^5 = 32$ along each axis) and a feature dimension of 256 features. This was flattened into a 1024-dimensional contextual information vector. We trained all models for 25 epochs.

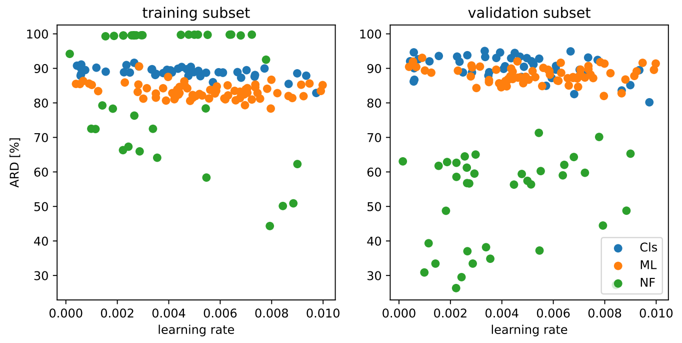

Fig. 9. The three different model training variants we evaluated: binary classification, metric learning and normalizing flows. We tested each variant with and without the optional context information. Binary classification and metric learning models were tested with our MLP architecture (Fig. 4) and our residual MLP architecture (Fig. 5). Each data point is the performance a model trained solely on the federal state Schleswig-Holstein using a random learning rate.

As can be seen in Fig. 9, normalizing flows are able to learn the training subset by heart but fail to generalize well to the validation subset. They also perform worse when adding context information. Both MLP-based model training variants performed well with a slight advantage to the binary classification-based approach. There is a slight increase in performance on the validation subset. For computing the validation performance we not only removed all cells belonging to training wind farms but also all cells within a $250m$ vicinity of those cells since we already noticed during development that models often tend to rate the immediate vicinity of existing wind farms very highly, i.e., they tend to suggest to simply

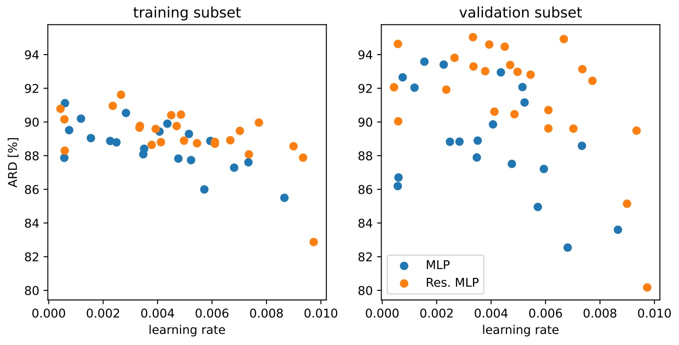

Fig. 10. Subset of the data shown in Fig. 9. Only binary classification data points are shown.

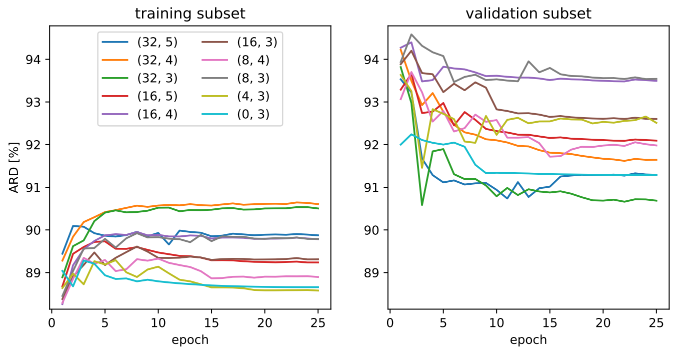

Fig. 11. Mean performance of the residual MLP model trained via binary classification. The configuration tuple specifies the extend of the context (first value) used in each cardinal direction and the number of downsampling steps used (second value). In the previous experiment (Figures 9 and 10) we only tested the configurations $(0, 3)$ and $(32, 5)$. The second value is irrelevant when using no context information (first value $= 0$).

increase existing wind farms instead of proposing new, well suited areas. Therefore, we remove the immediate vicinity of the training wind farms for validation purposes. As can be seen in Fig. 10, the residual MLP performed slightly better than the regular MLP. We therefore chose to focus on the residual MLP trained using binary classification as our best model variant from this point on.

We could not draw conclusions whether context information is actually helpful or not from the previous experiment. After deciding on the best model variant we ran another experiment in which we tested different context information configuration 15 times each. The mean performance across training epochs is shown in Fig. 11. When using less than the default five downsampling steps, we removed the later downsampling steps by setting the corresponding strides to 1 (from 2) while keeping the earlier downsampling steps. This is a common strategy also used in semantic segmentation models to increase spatial resolution (fewer downsampling steps) while keeping computational costs low (removing the late rather than early downsampling steps). While a large context of 32 helped with training performance, validation performance was actually best for the $(16, 4)$ and $(8, 3)$ configurations. Since there is no significant difference between those two configurations, we chose $(8, 3)$ as our best configuration going forward. The downward trend of the validation performance across the epochs already indicates that we train the models for too long, an issue we fixed by further hyperparameter optimization.

TABLE I

MEAN PERFORMANCE AND STANDARD DEVIATION OF THE BEST MODEL (RESIDUAL MLP; BINARY CLASSIFICATION; OPTIMIZEZD HYPERPARAMETERS) TRAINED AND EVALUATED ON DIFFERENT FEDERAL STATES. WE OMITTED THE CITY STATES BERLIN, BREMEN AND HAMBURG BECAUSE ALMOST NO LARGE ENOUGH WIND TURBINES HAVE BEEN COMMISSIONED IN THEIR AREA IN RECENT YEARS.

federal state	training performance (ARD)	validation performance (ARD)
Baden-Württemberg	$93.4\% \pm 0.5\%$	$77.4\% \pm 1.4\%$
Bayern	$81.2\% \pm 2.8\%$	$79.0\% \pm 4.1\%$
Brandenburg	$88.7\% \pm 0.5\%$	$86.7\% \pm 1.1\%$
Hessen	$90.3\% \pm 0.6\%$	$75.1\% \pm 2.5\%$
Mecklenburg-Vorpommern	$87.5\% \pm 0.7\%$	$83.5\% \pm 1.3\%$
Niedersachsen	$84.2\% \pm 0.9\%$	$86.1\% \pm 0.9\%$
Nordrhein-Westfalen	$88.9\% \pm 0.5\%$	$84.3\% \pm 0.7\%$
Rheinland-Pfalz	$89.9\% \pm 0.7\%$	$90.2\% \pm 1.0\%$
Saarland	$93.1\% \pm 1.1\%$	$87.8\% \pm 1.5\%$
Sachsen	$95.9\% \pm 0.8\%$	$77.1\% \pm 6.1\%$
Sachsen-Anhalt	$91.5\% \pm 0.6\%$	$92.6\% \pm 1.0\%$
Schleswig-Holstein	$89.1\% \pm 0.5\%$	$95.4\% \pm 0.4\%$
Thüringen	$94.7\% \pm 0.6\%$	$86.5\% \pm 1.7\%$
all	$86.1\% \pm 0.5\%$	$83.4\% \pm 0.7\%$

C. Best Model

After deciding on the best model variant, we ran a random search to optimize the hyperparameters used for our model. We used the federal state Schleswig-Holstein for this hyperparameter optimization process. Our optimized hyperparemeters are as follows. We set the number of input and output features of both linear layers in the residual blocks to 480 and reduced the number of blocks N to 7. The dropout probability was decreased to 0.025 as well. Furthermore did we change the number of features used by Xception. The model starts with 32 features after the first convolution and increases this number roughly by a factor of 2 until reaching the final number of features of 2048. We changed this to 24 after the first convolution and an increase by a factor of 1.5 up to the final number of features of 411. We used a learning rate of 0.0019 with the optimizer AdamW [26], [27] and cosine annealing learning rate schedule [28]. We trained for 14 epochs with a mini-batch size of 1024, half of which were postive samples and the other half were negative samples.

Performance results of the optimized model can be found in Tab. I. The model converges to good solutions, even in federal states with very few wind farms such as Saarland or Sachsen. The performance on the training subset ranges from 81.2% in Bayern to 95.9% in Sachsen. The validation performance ranges from 75.1% in Hessen to 95.4% in Schleswig-Holstein and is therefore, as expected, slightly worse than the training performance. While there still is some room for improvement in some federal states, the performance is already good enough to make a prototype as described in subsection III-D very viable and useful.

V. Conclusion

In this paper we presented WindGISKI, a project which aims to use AI to enhance a geographic information system to assist users in identifying areas suitable for the construction of new wind farms. We collected more than 60 geographical features for use in a large dataset covering Germany. We then conducted a survey among experts and used the results to identify a small subset of samples to use as positive and as negative samples for training a deep neural network model with residual connections. This model is able to assign suitability scores to every $50m \times 50m$ square in Germany. A work-in-progress prototype will make the AI's prediction accessible to end users to assist them in choosing suitable areas and help them understand why an area is considered suitable or not.

In future work we want to validate our AI model based on expert knowledge. The survey we conducted unfortunately had too few participants. We therefore are considering using statistical measurements which do not rely on absolute values such as the rank correlation for validation. A high rank correlation, i.e., the AI model and the experts rank feature importance and/or cell scores similarly, would point to the AI model actually reflecting expert knowledge. Another, more involved, approach could be to have experts manually score certain areas and compare their results to the AI's predictions. Or the experts could choose suitable areas from a larger region and, once the evolutionary algorithm part of our prototype is done, we could compare their choice to our prototypes optimization routine.

References

[1] Fachagentur Wind, "Hemmnisse beim ausbau der windenergie an land - ergebnisse einer branchenbefragung." https://www.fachagentur-windener-gie.de/fileadmin/files/Veroeffentlichungen/Genehmigung/FA_Wind_Ergebnisse_Branchenumfrage_06-2022.pdf, 2022. Accessed May 17th, 2024.

[2] Bund-Länder-Kooperationsausschusses, "Bericht des bund-länder-kooperationsausschusses zum stand des ausbaus der erneuerbaren energien sowie zu flächen, planungen und genehmigungen für die windenergienutzung an land, an die bundesregierung." https://www.bmwk.de/Redaktion/DE/Downloads/E/EEG-Kooperationsausschuss/2021/, 2021. Accessed May 17th, 2024.

[3] Fachagentur Wind, "Typische verfahrenslaufzeiten für genehmigung und realisierung, 30. wind-energietage 2022." https://www.fachagentur-windenergie.de/fileadmin/files/Veranstaltungen/2022-11-10_Kompetenztag_Windenergietage/FA_Wind_Kompetenztag_Verfahrenslaufzeiten_Quentin_10-11-2022.pdf, 2022. Accessed May 17th, 2024.

[4] R. Mari, L. Bottai, C. Busillo, F. Calastrini, B. Gozzini, and G. Gualtieri, "A gis-based interactive web decision support system for planning wind farms in tuscany (italy)," *Renewable Energy*, vol. 36, no. 2, pp. 754–763, 2011.

[5] M. A. Boggie, M. J. Butler, S. E. Sesnie, B. A. Millsap, D. R. Stewart, G. M. Harris, and J. C. Broska, "Forecasting suitable areas for wind turbine occurrence to proactively improve wildlife conservation," *Journal for Nature Conservation*, vol. 74, p. 126442, 2023.

[6] S. Grassi, F. Veronesi, R. Schenkel, C. Peier, J. Neukom, S. Volkwein, M. Raubal, and L. Hurni, "Mapping of the global wind energy potential using open source gis data," in *2nd Int. Conf. Energy Environ. bringing together Eng. Econ*, no. June, p. 6, 2015.

[7] P. Emami and A. Marzban, "The synergy of artificial intelligence (ai) and geographic information systems (gis) for enhanced disaster management: Opportunities and challenges," *Disaster Medicine and Public Health Preparedness*, pp. 1–3, 2023.

[8] S. R. Samaei and M. Ghahferokhi, "Ai-enhanced gis solutions for sustainable coastal management: Navigating erosion prediction and infrastructure resilience," in *2th International Conference on Creative achievements of architecture, urban planning, civil engineering and environment in the sustainable development of the Middle East*, 2023.

[9] M. Hamano, S. Shiozawa, S. Yamamoto, N. Suzuki, Y. Kitaki, and O. Watanabe, "Development of a method for detecting the planting and ridge areas in paddy fields using ai, gis, and precise dem," *Precision Agriculture*, vol. 24, no. 5, pp. 1862–1888, 2023.

[10] H. Ouchra, A. Belangour, and A. Erraissi, "An overview of geospatial artificial intelligence technologies for city planning and development," in *2023 Fifth International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, pp. 1–7, IEEE, 2023.

[11] S. Taşan, "Estimation of groundwater quality using an integration of water quality index, artificial intelligence methods and gis: Case study, central mediterranean region of turkey," *Applied Water Science*, vol. 13, no. 1, p. 15, 2023.

[12] R. Bill, J. Blankenbach, M. Breunig, J.-H. Haunert, C. Heipke, S. Herle, H.-G. Maas, H. Mayer, L. Meng, F. Rottensteiner, *et al.*, "Geospatial information research: state of the art, case studies and future perspectives," *PFG–Journal of Photogrammetry, Remote Sensing and Geoinformation Science*, vol. 90, no. 4, pp. 349–389, 2022.

[13] Z. Li and H. Ning, "Autonomous gis: the next-generation ai-powered gis," *International Journal of Digital Earth*, vol. 16, no. 2, pp. 4668–4686, 2023.

[14] R. Fabbri, L. D. F. Costa, J. C. Torelli, and O. M. Bruno, "2d euclidean distance transform algorithms: A comparative survey," *ACM Computing Surveys (CSUR)*, vol. 40, no. 1, pp. 1–44, 2008.

[15] T. Saito and J.-I. Toriwaki, "New algorithms for euclidean distance transformation of an n-dimensional digitized picture with applications," *Pattern recognition*, vol. 27, no. 11, pp. 1551–1565, 1994.

[16] M. Lindauer, K. Eggensperger, M. Feurer, A. Biedenkapp, D. Deng, C. Benjamins, T. Ruhkopf, R. Sass, and F. Hutter, "Smac3: A versatile bayesian optimization package for hyperparameter optimization," *Journal of Machine Learning Research*, vol. 23, no. 54, pp. 1–9, 2022.

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[18] O. et al., "Gpt-4 technical report," 2024.

[19] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain.," *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[20] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A convnet for the 2020s," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11976–11986, 2022.

[21] M.-H. Guo, C.-Z. Lu, Q. Hou, Z. Liu, M.-M. Cheng, and S.-M. Hu, "Segnext: Rethinking convolutional attention design for semantic segmentation," *Advances in Neural Information Processing Systems*, vol. 35, pp. 1140–1156, 2022.

[22] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[24] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1251–1258, 2017.

[25] L. Ardizzone, C. Lüth, J. Kruse, C. Rother, and U. Köthe, "Guided image generation with conditional invertible neural networks," *arXiv preprint arXiv:1907.02392*, 2019.

[26] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[27] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[28] I. Loshchilov and F. Hutter, "Sgdr: Stochastic gradient descent with warm restarts," *arXiv preprint arXiv:1608.03983*, 2016.