User-Centered Engineering of an Interactive Land Use Exploration Tool

Buhl, Tobias; Marcomin, David; Fallert, Stefan; Blechschmidt, Jana; Bönisch, Franziska; Mark, Robert; Cabral, Juliano; von Mammen, Sebastian

DOI:
10.2312/envirvis.20231109

Document Version
Publisher's PDF, also known as Version of record

Citation for published version (Harvard):
Buhl, T, Marcomin, D, Fallert, S, Blechschmidt, J, Bönisch, F, Mark, R, Cabral, J & von Mammen, S 2023, User-Centered Engineering of an Interactive Land Use Exploration Tool. in S Dutta, K Feige, K Rink & D Zeckzer (eds), *Workshop on Visualisation in Environmental Sciences (EnvirVis).* EnvirVis: Workshop on Visualisation in Environmental Sciences, The Eurographics Association, pp. 77-84, Workshop on Visualisation in Environmental Sciences (EnvirVis) 2023, Leipzig, Germany, 12/06/23. https://doi.org/10.2312/envirvis.20231109

Workshop on Visualisation in Environmental Sciences (EnvirVis) (2023)
S. Dutta, K. Feige, K. Rink, and D. Zeckzer (Editors)

User-Centered Engineering of an Interactive Land Use Exploration Tool

T. Buhl[1], D. Marcomin[1], S. Fallert[1], J. Blechschmidt[1], F. Bönisch[1], R. Mark[1], J. Sarmento Cabral[1,2] and S. v. Mammen[1]

[1]Julius-Maximilians-Universität, Würzburg, Germany
[2]School of Biosciences, University of Birmingham, Birmingham, United Kingdom

Abstract

In this paper we showcase a system for visualizing and predicting land-use data. The time series-based visualization application strives to improve science communication by facilitating the understanding of land-use change and is backed up by a machine learning-based land-use prediction application that imputes historic data and generates predictions of land use in the future. To present the project, we discuss the system's requirements which were developed by means of a User-Centered Engineering approach, elaborate on its current, early state of development and the corresponding results and finally discuss areas of potential improvement.

CCS Concepts

• ***Human-centered computing*** → ***Visualization; Human computer interaction (HCI);*** • ***Computing methodologies*** → ***Machine learning;***

1. Introduction

Biodiversity is an important part of the foundation for practically all aspects of human well-being, as it drives functional ecosystems in which people can thrive. Due to multiple human-induced threats including land-use changes, climate change, and air pollution, biodiversity is declining at an ever increasing rate [BSDN19, Sin02]. To counter this biodiversity crisis and its severe consequences, we have to understand how species react to human-induced threats. Based on this, we can evaluate which conservation measures can support the protection of biodiversity to which extent and prioritise accordingly. The underlying rationale and the conclusions also need to be communicated to the broad public to find the widespread political and societal support needed to put the identified measures into action. To this end, complex models need to be crafted that do not only integrate climate models but also consider the interplay between biodiversity and land use [CMPdS*22]. The general unavailability of models that can predict changes in land use in the long term poses a major challenge in this endeavour. While there is an increasing amount of satellite data available that classifies current and past land use, most land-use models only focus on the next few years [CMPdS*22]. The emerging availability, accessibility and accuracy of machine learning methods may help to increase the prediction range of these models. At the same time, such a data-driven approach may considerably shorten the time needed to develop a dedicated agent-based model for land-use prediction for a specific region [WBDD22]. Next to increasing the predictive capacities of land-use models, the models also have to be accessible so that they can help bridging the knowledge gap between domain experts and the broad public [Knu19]. To this end, several publications in the context of climate change research and public perception have shown that aesthetic and intuitive visualization can play a key role in effective science communication [NC05, She05]. However, suitable visualizations are still rare in land-use and biodiversity prediction.

In this paper, we present our efforts towards tackling both the challenges of future forecasting and interactive visualization. In particular, we have been following a User-Centered Engineering approach to create a software ecosystem that allows the users to navigate through time to explore the changes of land use. The project presented should fulfill two general goals. Firstly, it should allow for bridging the knowledge gap by providing a two-way visualization: a 2D-view providing high precision for those with scientific understanding of the matter to study a data set with and a 3D-view generalizing the presented data and facilitating its understanding for persons with little prior knowledge regarding the topic. Secondly, it should allow for machine learning-based generation of land-use predictions, thereby providing accurate and plausible data for scientific use, both performing data imputation on a time series and predicting potential land-use changes in the future. Additionally, the model should be adaptable to any region. Therefore, a dedicated model for land-use prediction for any region can be created with relative ease as long as there is enough data available.

In the remainder of the text, we will elaborate both on the project's requirements and the current state of its implementation, considering its workflow, its inner workings and the results it pro-

DOI: 10.2312/envirvis.20231109

vides. This specifies how the above goals should be realized in the final version as well as how they are realized as of now. Finally, we will outline future work, especially also in light of the early state of the presented proof-of-concept implementation.

2. Related Work

Before diving into the specifics of the project itself, we discuss some similar works which have served as both inspiration and guidance while developing it. Firstly, the idea of predicting data using some form of model and then visualizing it to facilitate understanding is by no means new. For instance, Copernicus, the Earth observation program of the European Union, provides a website on which the user can study the land cover, i.e. the physically observed coverage, of various European countries and its changes over time. The website allows the user to step from the year 2000 to 2018 in increments of 6 years and provides a map that shows the density of some types of land cover such as artificial surfaces or forests and semi-natural areas. Additionally, the map shown is supplemented with graphs on the land cover distribution and its relative change over time. The solution can be found on [Cop23b]. While the website provided by the Copernicus project is an example for an existing solution regarding the visualization application, related solutions for the prediction application also exist. For example, the SLEUTH land use change model uses a cellular automata based approach to predict future land use requiring slope, land cover, excluded, urban, transportation and hillshade images [CC13]. However, in contrast to SLEUTH, our implementation strives to work optimally while only using land cover data as input and does not make use of cellular automata, using a machine learning model for direct prediction instead. A general overview of the current state of land-use prediction models as well as their characteristics, capabilities and limitations is provided in [WBDD22]. Furthermore, the authors conclude that machine learning has strong potential, giving inspiration for our specific approach to the prediction. In [Knu19] the general necessity of scientific communication regarding topics such as global warming and climate change is stressed, as while the number of peer reviewed publications on these topics has rapidly increased in recent years, the corresponding actions, e.g. reduction of CO_2 emissions from fossil fuels, have not. This can be attributed to a knowledge gap between domain experts and the broad public. Finally, in [She05], the need for visualizations like ours is highlighted very well. To properly communicate climate change, its extent and its issues to the public, numbers do not suffice. Instead, images and interactive visualization are needed, e.g. of land-use change.

3. Requirements

As our project is not supposed to be a simple tech demo, but rather provide real world use, we have been following the User-Centered Engineering approach [RF14]. The two stakeholders in the project and co-authors of this paper, Jana Blechschmidt and Stefan Fallert, are researchers working at the Center for Theoretical and Computational Biology of the University Würzburg. They focus on biodiversity and how it is impacted by climate and land-use change. They introduced the need for a software that allows for a visually appealing representation and navigation of geographical data over time. As many of these data sets contain missing years, it was clear from the start that a machine learning model would have to perform data imputation and also make predictions about future changes. These high-level user requirements led to a vast number of system and, in consequence, functional requirement formulations which served as the basis for developing out system. In the following, we briefly extend those inferred requirements that had the greatest impact on our system design and implementation. In the end, our application allows to pre-process, train and interactively visualize land-use data of a given geographic region. An impression of the application which was developed from these requirements can be seen in Figure 1.

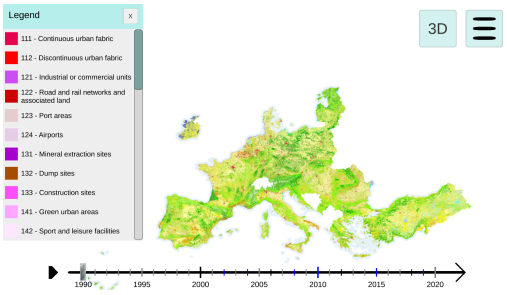


Figure 1: *The interactive visualization application. The legend is currently shown and the selected year is 1990. A button to toggle between 2D-view and 3D-view as well as a sandwich menu for import/export and changing the navigation related keybinds is in the upper right corner. The timeline, which can be used either for scrubbing or selecting a specific year, is at the bottom.*

The prediction application needs to be able to import land-use raster data, generate data for the missing years as well as potential future predictions with a properly trained model and export the results in a format usable for scientific purposes even outside the scope of this project. After the data generation, the visualization application needs to import the land-use raster data files and discern what is generated and what is real. Thereafter, the data has to be visualized in an easy-to-understand and nice-to-look at way. This has to include a timeline to observe changes between the years as well as basic navigation options.

The User-Centered Engineering methodology yields the outlined requirements and also anticipates that they are fulfilled once the expectations of the stakeholders are met and according tests are passed. As a starting point we interviewed the stakeholders to listen to their idea of the project. This revealed the broad user requirements stated earlier. Afterwards, to prepare for the refinement process, we considered existing solutions and literature on the topic in a corresponding market and research survey. Next, we performed a cognitive walk-through which means that we established personas who represented potential users and then inferred use cases. Considering the use cases, we were able to deduce system and functional requirements and to create a paper prototype of how our applications could potentially look. These non-functional deliverables provided the basis of a first feedback cycle with the stakeholders. In each cycle, the current results are presented to the stakeholders and feedback is inquired in a semi-structured interview. Next, the

feedback is conceptually integrated both in a formal requirements graph and the implementation concept, and realized. Each cycle happens over the course of about one month. Following this approach, we arrived at a set of continually refined functional/design requirements that we detail below.

The import system needs to identify GeoTiff files, i.e. image data augmented with real-world georeferences, in a given folder and recognize which year of data they encode, as well as whether it is captured real-world or generated data. A supervisor needs to verify this data or provide it, in case no hints to the time period were automatically recognized. Next, the GeoTiff data is converted to mathematical objects or bitmaps for prediction and visualization, respectively. For the prediction application it is further required to split the input into geographically associated, time-stamped tiles to facilitate the machine learning process. To adapt the prediction capabilities to a specific use case, the user should be empowered to configure the machine learning model, e.g. to specify the maximum number of iterations or to provide a custom evaluation metric. Next, a model can be trained by means of a suitable, external machine learning library. The resulting, trained model should then be saved to disk and can be used for data completion and prediction. The data prediction and completion system needs to allow for the user to specify one or more GeoTiff files to use as the basis as well as the years they need filled and the model trained beforehand. Optionally, the user should be able to specify a part of the data set they want a prediction for, instead of generating a prediction for the whole set. The process should run automatically by loading the model and performing the appropriate predictions for each pixel in the raster data and saving the results as a Tiff file. In the end, additional data such as the geotransform, i.e. the georeference data associated to the data set, should be copied over with adjustments where needed from the GeoTiff files provided. After obtaining a time series of generated and real data sets and successfully importing them, we can begin using the interactive visualization application. The land-use rendering engine needs to take the bitmap from the pre-processing pipeline as an input and display it in an attractive and informative way. To connect contextual with detailed data, an option to switch between 2D-view and 3D-view should be offered. As the rendering engine works together with the timeline, it is important to reduce loading times between the time steps the user can navigate across, e.g. by keeping as much data as possible in memory, ready to be displayed at any time, and only showing high details where needed. To further improve navigation, a camera control system needs to offer ways of moving the camera and zooming into certain areas to explore them in more detail. The timeline system, on the other hand, should provide both options for displaying a specific year as well as scrubbing across years forward or backward in time. Finally, a legend needs to be provided as part of the user interface system. The legend displays the different land-use classifications of the data set and can be shown or hidden to not disturb the user's exploration. Optionally, when selecting a section of the map, its classification should be highlighted in the legend. In this way, the user can easily understand what type of land use they are looking at, whenever the visualization does not convey it sufficiently already.

4. Engineered Solution

In the following section, we present the concept and implementation that has emerged as part of the User-Centered Engineering process. In particular, we present an overview of the solution we are working on, dive into its system components and relate their features to the captured and continuously refined requirements.

4.1. System Overview

The general workflow of the system is split into two applications, one to interactively visualize data and one to pre-process and generate data at new time steps based on machine learning. For details on the first application refer to Figure 1. Meanwhile, the latter is used by entering file paths and parameters for training and predictions through a command line interface.

Next, before elaborating on the system's components, we consider the data set it is being developed for. While it should perform on a multitude of data sets, we specifically consider the Corine Land Cover [Büt14], as per suggestion of the stakeholders. It contains geographical raster data in a 100 meter resolution with a minimum mapping unit of 25 hectares for areal phenomena, provided as GeoTiff files with a time series featuring the years 1990, 2000, 2006, 2012 and 2018. The data set provides information on the biophysical characteristics of the Earth's surface by categorizing each pixel in the raster as one of 44 classes. The total size of each GeoTiff in the data set is 46,000 by 65,000 pixels. The minimum amount of data to train the tool is technically two time steps with every pixel classification appearing at least once in one of the two sets. However, not using the entire Corine Land Cover has produced subpar results in our experience. For more information on who produces and coordinates this data set and how it is made, consider visiting their website at [Cop23a].

4.2. Interactive Visualization

The interactive visualization application is being developed in the Unity Engine (s. https://unity.com). For details on its user interfacerefer to Figure 1. To start using the visualization application, the user first needs to import the relevant data sets. The import system requires the user to provide a list of file paths paired with the corresponding years linking to the data sets they want to load, as well as information whether the data sets contain generated or captured real-world data. Before any visuals can be generated, the legend needs to be created first. To do so, the corresponding QML file for each data set, i.e. a file containing the list of classifications and the colors and pixel values associated with them, is opened and the name and color values for each classification are extracted. The legend generated this way is displayed as a hideable scroll view, as to not obstruct the user's vision during exploration, and also saved internally as it is key for visual generation later on. Now, we can use the legend as well as the file paths to GeoTiff files and additional information to prepare generating the visuals. First, the data formatting system opens the GeoTiff file as an array using GDAL (s. https://gdal.org) and compresses the result by a factor of n using built-in functions. The default value for n is 10, but it can be changed by the user. This is both to ensure that the system does not run out of memory and to guarantee fast loading times of the

resulting images during use. The decrease in resolution is not a major concern in most cases, as the minimum mapping unit for areal phenomena in the Corine Land Cover is larger than the resolution of the GeoTiffs, as mentioned earlier. Next, a bitmap is generated pixel by pixel using the data in the now compressed array. The color assignment used in the generation is the one determined for the legend earlier. Finally, the bitmap is saved to disk as a JPEG file with low compression. The filename denotes the year it is assigned to and whether it contains generated data or not.

Now, that all data is imported and everything is processed, the user can start exploring. The application at this point in time can be seen in 1. To navigate the displayed map simply use the WASD keys as well as the mouse wheel for zooming in and out. It was determined, that navigation buttons would obstruct the view too much, therefore keybinds were used instead. The system behind this works by moving and scaling the map, which is internally handled as part of the user interface. Furthermore, the timeline system is implemented such that pressing the play button next to the timeline results in the years being scrubbed slowly forward in time with the currently selected year as the starting point. To do so, the cursor is advanced to the corresponding spot and the visual generation system is notified that is must now load the correct year. Should no data for a year be available, the next oldest available data is used. The default selected year upon opening the application is the first year available in the timeline. The user can also manually select a tick on the timeline to jump to that year. The ticks are generated by considering the first and last year available in the data sets provided and generating a big tick with a year label every 5 years and a small tick every year that does not have a big one with all ticks, no matter if big or small, being clickable. Furthermore, years with no data set available have their tick grayed out and years with a generated data set are marked with a blue tick. Additionally, the user can also hide and show the legend by clicking the x-button or clicking the legend icon-button respectively. Finally, the user interface systems offers an icon button for switching between 2D- and 3D-view as well as a sandwich menu for import of GeoTiff files, export of generated JPEG files and options, e.g. for changing keybinds. Both of these are non-functional as of now.

Behind the scenes, the visualization is handled by the land-use rendering engine. At any time, a year can be requested to the engine for loading. In response, the corresponding JPEG file is opened and applied as a texture to the map UI element. As the 3D-view is not implemented yet, there is also no need for a more sophisticated visual generation system. On the contrary, the current simple system ensures high efficiency and keeps loading times to unnoticeable amounts, allowing for smooth exploration of the time series.

4.3. Machine Learning Model

Before examining the prediction application, we first discuss the machine learning model behind it. We use LightGBM, an open-source gradient boosted decision tree library developed by Microsoft. It takes a list of features as its input and returns a single classification as its output. LightGBM uses histogram-based algorithms, which are both faster and more memory efficient than the typically used pre-sort-based algorithms (s. https://lightgbm.readthedocs.io/en/latest/Features.html) and also compare favorably with the competition [KMF*17]. We chose it specifically over other libraries for both is speed and memory efficiency as well as ease of use. In our implementation the features are the classifications of a chosen pixel in the raster data as well as a certain amount of pixels surrounding it together with a number representing how many years into the future the model should predict. The resulting classification returned by the model is the predicted classification of the chosen pixel that many years into the future.

Decision tree learning is a form of supervised learning that uses trees, i.e. refer to Figure 2, which classify instances by sorting them based on feature values. Each node in the tree represents a feature of the instance which is to be looked at for classification. Each branch from that node represents the values that feature can potentially have. The leaves of the tree represent the final classification [KZP*07].

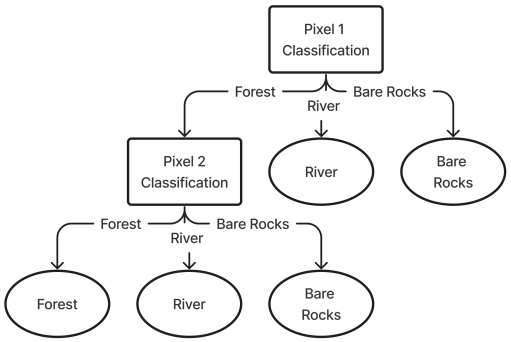


Figure 2: *A simple example of a decision tree. Adapted from [KZP*07] to fit our application domain.*

The difficulty now lies in constructing an optimal decision tree such that there is as little error as possible. To do so, a weak model and a strong model are used. The strong model is improved on until it becomes a suitable decision tree for the classification task. The weak model is trained to predict the gradient of the loss of the strong model's output, i.e. the error of the strong model. Finally, the weak model is subtracted from the strong model to produce a new strong model, which now has a smaller error than the previous one [Goo23]. This is repeated until either a satisfying accuracy is reached, accuracy gain is stagnating or a predetermined number of iterations is over. Our solution relies on a set number of iterations.

4.4. Prediction

With the context of the machine learning model used in mind, let us elaborate on the prediction application. It is written in python and based on converting GeoTiff files into a mathematical structure the model can learn and work with. Specifically, we chose numpy matrices (s. https://numpy.org). These objects are then used to prepare training data for the model. After the model has finished learning, it can be used to generate predictions for any GeoTiff file provided. Results from this prediction are saved as GeoTiff files themselves. The corresponding workflow is demonstrated in Figure 3. Internally, this entire process happens in multiple steps.

First, the application identifies all GeoTiff files within a given

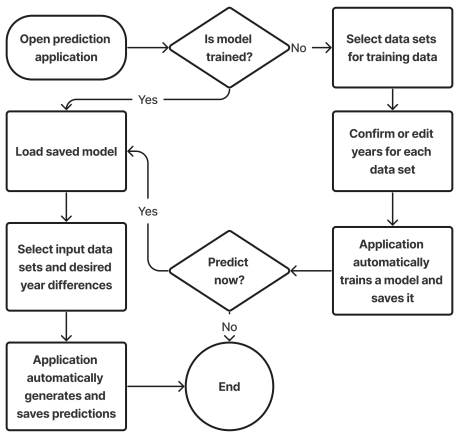


Figure 3: *A workflow diagram for the prediction application. The user has the options of loading a model or training a new one.*

folder and all of its sub-folders. Afterwards, it guesses the corresponding years based on filename, prints them on a command line interface and asks the user for correction. To format the data, the GeoTiff files are then opened with GDAL and converted into numpy matrices for later use. Second, the application partitions the data sets of each year into $n \times n$ pixel sized tiles and saves them with their respective coordinate. This is done, because the model uses an input of $n^2 + 1$ features, consisting of the pixels in a tile as well as the desired year difference between the input and the output, i.e. the predicted classification of the tile's central pixel in the future. Currently the number n is hard-coded to be 125, as we found this to work rather well. This splitting process is an important extra step taken for the sake of memory efficiency. Third, the application generates training data for the model. It does so by generating a list of all tile coordinates that do not contain a tile filled with only no-data classified pixels at any year. Thereafter, that list is shuffled and cut off at a point such that training data generation does not use more than a given amount of memory. Afterwards, for each tile coordinate and combination of two years an instance is created, the features of which are the classifications of the pixels in the tile at the time of the earlier year as well as the time difference between the years. Additionally, each instance is given a label, namely the classification of the tile's center pixel at the time of the newer year. These instances are compiled into a LightGBM data set and saved as a binary for easy loading. Next, the model is trained using the generated training data. This works automatically thanks to the LightGBM library. Afterwards it is saved and can be loaded for reuse at any time. Lastly, the prediction itself happens. To predict, the input of a GeoTiff file and a year difference as well as a trained model are required. The GeoTiff file is opened with GDAL and converted into a numpy matrix and the model is loaded. Thereafter, for each pixel in the matrix, a $n \times n$ pixel area around that pixel is considered, given to the model and the result is written in a new matrix at the same coordinate. If the $n \times n$ area is partially outside of the provided matrix, the outside portion will be filled with no-data classified pixels. As a notable optimization, only non-no-data pixels are considered for the aforementioned process. No-data pixels are instead immediately copied over into the results matrix. The result of running this process for each pixel is a new matrix of the same shape as the input generated from the provided GeoTiff file. This new matrix contains all classifications generated by the model in the correct arrangement. Finally, it is converted back into a GeoTiff file and data such as the geotransform is copied over from the original.

5. Results & Discussion

In this section, we discuss the current state of the interactive application, including its prediction engine. As explained earlier, the User-Centered Engineering approach implements the principle that a solution is accepted, if it fulfills the expectations of the stakeholders and is passing according tests. Therefore, we first consider the current evaluation provided by the stakeholders. We specifically inquired for evaluative feedback at this point and it was provided by the stakeholders in written form.

> There are currently two fully functional prototypes for the interactive visualization part and the data-driven generation component. They implement the most important requirements and proof the feasibility of the project's goal and scope. The presented user interface with the options to scroll through the timeline and to visualize the changes in the landscape in real time works intuitively even for inexperienced users. Especially the option to switch between the 3D and 2D view is a very useful tool to allow looking at the data both in a scientific way, i.e. by provision of the 2D overview with a clear legend and easily discernible contextual as well as detail views, and in a more communication-focused, aesthetically pleasing way by means of the 3D rendering options. In this context, the zoom functionality is highly useful, allowing for deepened investigation of local and regional changes.
> At this point, the process of loading and exporting the data is adequate for an experienced, scientifically trained user but could be improved by reducing the number of required, manually conducted steps and thereby generally providing better usability. It could further be improved by supporting greater variations of data formats, ideally with an editor for data layout schemes. To reach out to the broad public, the software needs to become more readily accessible, e.g. by provision of a common installation wrapper or a web front end. The performance of the machine learning methods seems promising, but their predictive performance has yet to be rigorously evaluated.
> Overall, the project has been advancing rapidly and has been addressing our needs and feedback. Even at its current state, the implemented solutions may already prove useful for our research by imputing historic land-use data sets and by the provision of an impressive visualization of the changes that the land use undergoes over time.

Next, we have to consider proper tests according to these expec-

tations and evaluate how well the current state of the project passes them, both for the visualization and the prediction application. For the first, we compare the achieved solutions with the collected set of requirements. For the latter, we consider three different evaluation metrics: accuracy and speed of the prediction as well as the contextual plausibility of the generated land-use maps.

Let us begin by diving into the comparison of the current state of the visualization application to its requirements. Firstly, the input system fulfills its functional requirements. However, it can be criticised that the time taken by the data formatting system to generate the bitmaps can be rather long for large data sets. Furthermore, it is currently not properly coupled with the user interface as both the menus for importing a data set and exporting the generated image are non-functional. Next, the navigation system does its job according to the system requirements and is positively evaluated by the stakeholders. Both zoom and movement are supported in the 2D-view. The same goes for the timeline system, which supports the key features year selection and scrubbing while also displaying the available and generated years in the time series clearly. Notably, the year changes can be viewed smoothly without any lag. Furthermore, the legend is displayed properly and can be hidden. It can be noted that optional features regarding the legend, such as highlighting the corresponding classification in the legend when selecting an area of the map, are yet to be implemented. The sandwich menu for options and import/export works as intended visually. Additionally, some improvements to the visual appeal of the user interface can be made, adjusting it to follow more modern style guides. Lastly, the land-use rendering engine is to be considered. It fulfills the functional requirements regarding the 2D-view as it is both fast and efficient. However, the JPEG method used sometimes provides subpar results when zooming in very closely. Additionally, it can be noted that the lack of a map showing locations of country borders and important cities can hinder the immediate understanding of the context of the data displayed. The main critique lies in the complete lack of an implementation of the 3D-view. While it is currently in a stage of being prototyped on paper and conceptually developed, it is entirely absent from the actual application as of right now.

Next, let us move on to the results of the prediction application. We begin by defining a metric for its accuracy. The accuracy of the model represents the percentage of correctly classified test cases. To calculate it, sets of validation data are constructed alongside the training data. While these sets of validation data are of the same format as the training data, they are generated independently to more properly reflect the models performance in a real use case. Ground truth is then established by observing the real world data provided by the Corine Land Cover during the next time step. This is compared to the models prediction on the corresponding input the same amount of time into the future. The median accuracy of the model with 10 iterations of training is about 82% with the highest being 82.82%. After that number of iterations, performance increase becomes marginal. Therefore, the 10 iterations model is used for the rest of this section. Next, to eliminate the possibility of the model predicting everything correctly by chance, we consider Cohen's Kappa. It is calculated as follows [Coh60]:

$$\kappa = \frac{p_o - p_e}{1 - p_e}$$

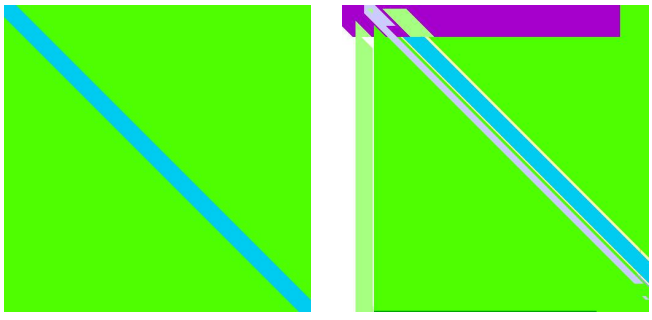
Here, p_o refers to the observed inter-rater agreement, i.e. the percentage of test cases both raters arrive at the same conclusion for. In our case this is equivalent to the accuracy of the system, as the raters are the ground truth on the one hand and the classification predicted by the model on the other hand. Furthermore, p_e refers to the probability of agreement by chance. It is calculated as follows, where N is the number of test cases, n_{ki} refers to the amounts rater i predicts category k and c is the total number of categories:

$$p_e = \frac{1}{N^2} \sum_{k=1}^{c} n_{k1} n_{k2}$$

As a result, p_e calculates to about 10% and we obtain a Cohen's Kappa of about 0.8. This is generally accepted as strong agreement [McH12]. Therefore, we can conclude that, mathematically speaking, the accuracy of our model is rather satisfying.

Next, we examine the speed of the prediction by looking at the time it takes to predict a 500×500 pixel GeoTiff. However, as we optimized our prediction algorithm to skip no-data pixels, we need to consider two different benchmarks. One, for the time it takes to predict a 500×500 pixel GeoTiff full of actual data, and another one, for one of the same size only containing no-data pixels. These times average out as 90 - 100 seconds for full GeoTiffs and 0.1 - 0.2 seconds for empty GeoTiffs over a multitude of test runs. While this does not sound very slow at first, this scales very badly for large data sets such as the Corine Land Cover. We can conclude, that the prediction speed is in dire need of improvement to drastically improve the user experience.

Finally, we consider the plausibility of the prediction. To do so, we have set up three test cases. Firstly, we simply look at an area of bare rocks that also goes on beyond the edges. Secondly, we consider a square of forest with a river diagonally flowing through it. Finally, we take a random section of the Corine Land Cover. For all of these we discuss the plausibility of the prediction generated by the model 6 years into the future. Unsurprisingly, the area of bare rocks is predicted to stay an area of bare rocks with not much interesting to discuss. This confirms that the model is at least capable of the most basic of tasks. Next, let us take a closer look at Figure 4 featuring an area of mixed forest with a water course crossing it diagonally. It is rendered using the 2D-view of the visualization application. For a detailed legend, please refer to the one provided by [Cop23c].

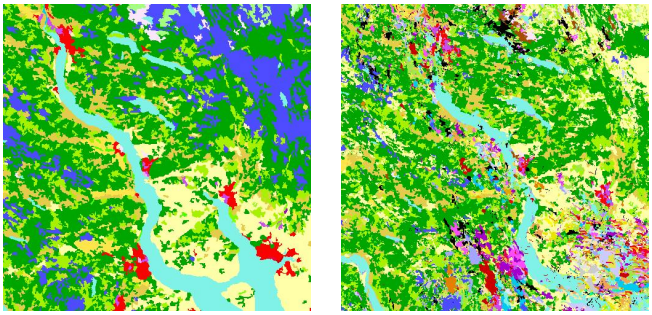


(a) (b)

Figure 4: *(a) shows the input of a simplified river flowing throughout mixed forests, (b) the prediction generated by our model for that area 6 years into the future.*

Notably, the edges of the image appear to be nonsensical, con-

taining classifications such as mineral extraction sites, no-data and moors and heathland. This is due to the model having to work with large portions of no-data pixels in its input in those areas. Additionally, south of the river a diagonal line of salt marshes can be seen and the river itself has inexplicably moved north. Now, before drawing conclusions from this, let us take a look at the prediction of the section of the Corine Land Cover presented in Figure 5.



(a) (b)

Figure 5: *(a) shows the original section of the Corine Land Cover, (b) the prediction generated by our model for that same section 6 years into the future.*

We can again see a similar trend. While the general concept of a river flowing through a forest-rich environment with some urban fabric along its course has been maintained, the details have been warped. The shape of the river has mostly been maintained, though its thickness is reduced. Furthermore, the various kinds of urban fabric have spread out and the peat bogs have largely been replaced by forests. We can conclude that the plausibility is not quite satisfying yet and especially struggles as soon as there are large amounts of no-data pixels, such as around the edges of a GeoTiff file. However, it is headed in the right direction and through proper adjustment of the training data, we assume it is possible to improve plausibility and mathematical accuracy.

6. Next Steps

The immediately necessary work we can deduce from the results we shared mainly belongs to two categories: (1) The implementation of missing features and (2) the improvement of prominent issues in the current state of the application. Firstly, an issue found in both applications is a lack of speed of both the import and prediction component. This can be addressed by optimizing the corresponding code segments, particularly with paying attention to pruning unnecessary calculations, on-demand processing, refinement of the implemented algorithms, and optimal use of externally provided functionalities. Secondly, some features important to the general interaction loop are missing from both applications. This includes the lack of easy import and export options in case of the visualization and a way of configuring the model through the command line interface in case of the prediction component. The resulting implementation works afford the pursuit of engineering rather than research methods, and can be well-planned for the near future. Additionally, there are some user experience issues which require our attention. This is mainly in reference to loading and exporting the required data, both in regards to the number of manual steps needed and their complexity. It also includes issues such as the lack of a map to contextualize the visualized data with. Furthermore, regarding the plausibility of the prediction application, better methods of generating proper training data and better calibrations for the model need to be found through trial and error. Finally, the interactive visualization application still needs to develop its 3D-view which is a central selling point of the project. Fortunately, a lot of conceptual work has already been done. However, there are still open questions regarding its implementation. Optionally, support for different file formats and types of data sets can also be considered as future work. After these main categories of issues have been addressed, the final task for completion of the implementation work is merging the two applications into a single out-of-the-box project and its public release.

7. Conclusion & Future Work

Despite the apparent need for further development steps, the presented proof-of-concept implementations already facilitate both science and science communication of land-use change. They allow one to interactively explore the evolution of land use over time, backed by a machine learning model that imputes historic data and generates predictions for the future.

In terms of future work, we envision various directions the presented project could support. For instance, the resulting tool could be tailored to different user groups and be actively deployed, for instance, in school curricula, be adjusted to support decision makers that need to weigh the pros and cons of regional developments, or provide a platform for human factor studies investigating interactive visual analytics mechanisms, or for experimenting and evolving time series predicting machine learning approaches.

References

[BSDN19] BRONDIZIO E. S., SETTELE J., DÍAZ S., NGO H. T.: Global assessment report on biodiversity and ecosystem services of the intergovernmental science-policy platform on biodiversity and ecosystem services. *IPBES* (2019). 1

[Büt14] BÜTTNER G.: Corine land cover and land cover change products. *Land use and land cover mapping in Europe: practices & trends* (2014), 55–74. 3

[CC13] CHAUDHURI G., CLARKE K.: The sleuth land use change model: A review. *Environmental Resources Research 1*, 1 (2013), 88–105. 2

[CMPdS*22] CABRAL J. S., MENDOZA-PONCE A., DA SILVA A. P., OBERPRILLER J., MIMET A., KIESLINGER J., BERGER T., BLECHSCHMIDT J., BRÖNNER M., CLASSEN A., ET AL.: The road to integrate climate change effects on land-use change in regional biodiversity models. *Authorea Preprints* (2022). 1

[Coh60] COHEN J.: A coefficient of agreement for nominal scales. *Educational and psychological measurement 20*, 1 (1960), 37–46. 6

[Cop23a] COPERNICUS PROGRAMME: Corine land cover, Last accessed 06. March, 2023. URL: https://land.copernicus.eu/pan-european/corine-land-cover. 3

[Cop23b] COPERNICUS PROGRAMME: Land cover and land cover changes in european countries in 2000-2018, Last accessed 06. March, 2023. URL: https://land.copernicus.eu/dashboards/clc-clcc-2000-2018. 2

[Cop23c] COPERNICUS PROGRAMME: Clc 2018, Last accessed 08. March, 2023. URL: https://land.copernicus.eu/pan-european/corine-land-cover/clc2018. 6

[Goo23] GOOGLE DEVELOPERS: Gradient boosted decision trees, Last accessed 06. March, 2023. URL: https://developers.google.com/machine-learning/decision-forests/intro-to-gbdt. 4

[KMF*17] KE G., MENG Q., FINLEY T., WANG T., CHEN W., MA W., YE Q., LIU T.-Y.: Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems 30* (2017). 4

[Knu19] KNUTTI R.: Closing the knowledge-action gap in climate change. *One Earth 1*, 1 (2019), 21–23. 1, 2

[KZP*07] KOTSIANTIS S. B., ZAHARAKIS I., PINTELAS P., ET AL.: Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering 160*, 1 (2007), 3–24. 4

[McH12] MCHUGH M. L.: Interrater reliability: the kappa statistic. *Biochemia medica 22*, 3 (2012), 276–282. 6

[NC05] NICHOLSON-COLE S. A.: Representing climate change futures: a critique on the use of images for visual communication. *Computers, environment and urban systems 29*, 3 (2005), 255–273. 1

[RF14] RICHTER M., FLÜCKIGER M.: User-centred engineering. In *User-Centred Engineering*. Springer, 2014, pp. 11–24. 2

[She05] SHEPPARD S. R.: Landscape visualisation and climate change: the potential for influencing perceptions and behaviour. *Environmental science & policy 8*, 6 (2005), 637–654. 1, 2

[Sin02] SINGH J.: The biodiversity crisis: a multifaceted review. *Current Science* (2002), 638–647. 1

[WBDD22] WANG J., BRETZ M., DEWAN M. A. A., DELAVAR M. A.: Machine learning in modelling land-use and land cover-change (lulcc): Current status, challenges and prospects. *Science of The Total Environment* (2022), 153559. 1, 2