The Error is the Feature: How to Forecast Lightning using a Model Prediction Error

Christian Schön
Saarland Informatics Campus
Big Data Analytics Group

Jens Dittrich
Saarland Informatics Campus
Big Data Analytics Group

Richard Müller
German Meteorological Service
Offenbach, Germany

ABSTRACT

Despite the progress within the last decades, weather forecasting is still a challenging and computationally expensive task. Current satellite-based approaches to predict thunderstorms are usually based on the analysis of the observed brightness temperatures in different spectral channels and emit a warning if a critical threshold is reached. Recent progress in data science however demonstrates that machine learning can be successfully applied to many research fields in science, especially in areas dealing with large datasets. We therefore present a new approach to the problem of predicting thunderstorms based on machine learning. The core idea of our work is to use the error of two-dimensional optical flow algorithms applied to images of meteorological satellites as a feature for machine learning models. We interpret that optical flow error as an indication of convection potentially leading to thunderstorms and lightning. To factor in spatial proximity we use various manual convolution steps. We also consider effects such as the time of day or the geographic location. We train different tree classifier models as well as a neural network to predict lightning within the next few hours (called *nowcasting* in meteorology) based on these features. In our evaluation section we compare the predictive power of the different models and the impact of different features on the classification result. Our results show a high accuracy of 96% for predictions over the next 15 minutes which slightly decreases with increasing forecast period but still remains above 83% for forecasts of up to five hours. The high false positive rate of nearly 6% however needs further investigation to allow for an operational use of our approach.

CCS CONCEPTS

• **Computing methodologies** → *Classification and regression trees*; *Neural networks*; • **Applied computing** → *Earth and atmospheric sciences*.

KEYWORDS

tree classifier; neural networks; lightning prediction; satellite images; optical flow; machine learning; meteorology

1 INTRODUCTION

Weather forecasting is a very complex and challenging task requiring extremely complex models running on large supercomputers. Besides delivering forecasts for variables such as the temperature, one key task for meteorological services is the detection and prediction of severe weather conditions. Thunderstorms are one such phenomenon often accompanied by heavy rain fall, hail, and strong wind. However, predicting them and giving precise information about their severity and moving direction is a hard task.

Current state-of-the-art systems such as *NowCastMIX* [15], a system operated by the Deutscher Wetterdienst, combine a multitude of data sources to generate warnings with a spatial resolution of 1×1 km and a temporal resolution of five minutes for severe weather conditions such as thunderstorms. Brightness temperatures of spectral satellite channels and their differences exceeding a certain threshold are interpreted as a sign for critical conditions potentially leading to thunderstorms. Radar systems are used to detect water particles in the atmosphere and clouds potentially developing towards thunderstorms whereas numerical weather prediction (NWP) models offer estimations of the near storm environment. Lightning detection systems allow us to localize thunderstorms by measuring electric waves. Even using such advanced models, the prediction of thunderstorms remains very challenging, especially for forecast periods larger than one hour when the false alarm ratio increases to more than 80%. The key to thunderstorm forecasting is the early and precise detection of convection.

Satellite data, which is nowadays part of many weather forecasting products and offered a significant performance boost for larger forecast periods, however is not yet established for the operational prediction of developing thunderstorms even though it offers a combination of high spatial and temporal resolution, two key elements for a successful forecast. Our approach therefore investigates a new way to predict thunderstorm clouds based on satellite data. The core idea of our work is to use the prediction error of a first prediction model as a feature. That error-feature is then used for the actual, second, different prediction model. In more detail, we compute the error of two-dimensional optical flow algorithms applied to images of meteorological satellites and interpret it as an indication of possible movement in the third dimension (convection) potentially leading to thunderstorms and lightning.

Our main contributions to the problem of forecasting thunderstorms are the following:

(1) We present a prediction model to forecast lightning using satellite images. The core idea of our work is to use the prediction error of a first prediction model as a feature for a second (different) prediction model. Although learning from the error of a prediction model reminds of boosting, our approach differs from such methods as described in Subsection 4.1.

(2) We present a new set of features based on the error of optical flow algorithms applied to satellite images in different channels which can be used to predict thunderstorms. These features differ from previously used approaches mainly based on temperature differences as presented in Section 2. Based on this feature set, we then apply different machine learning algorithms in order to automatically predict the occurrence of lightning. This process is described in more detail in Section 4.

(3) An evaluation of these error-based features used to predict the immediate future, i.e. the next 15 minutes, show the importance of convolution, especially with large kernel sizes. This evaluation also motivates the use of a broad range of channels. A more detailed description of the results can be found in Section 7.

(4) We present experiments showing the capability of machine learning models to predict lightning based on features derived from the error of optical flow algorithms as well as additional, satellite-based features on a balanced test set containing lightning with a forecast period of 15 minutes up to 5 hours. The results show an accuracy of more than 96% for the best model on the smallest period, decreasing to 83% for the largest period.

This paper is structured as follows: Section 2 gives a quick recap of the physics of thunderstorms followed by a review of current forecasting systems in Section 3. We introduce our new approach in Section 4, followed by the necessary data preparation and feature generation steps presented in Section 5. Section 6 briefly introduces the experimental setup before we present the first set of experiments and results in Section 7 based on the approach described before. Section 8 introduces additional features beyond pure error-based attributes and evaluates the impact of this enhancement compared to the original approach. In Section 9, we present the performance of our approach for larger forecast periods.

2 A QUICK RECAP OF THUNDERSTORMS

In this section, we briefly review the physics of thunderstorms. Thunderstorms belong to a class of weather phenomena called *convective systems*, characterized by an updraft of warm air from lower levels of the atmosphere towards higher and colder levels, accompanied by a downdraft allowing the cool air to flow back towards the ground. Thunderstorms essentially form a special case of convective systems which are characterized by a strong updraft of warm, moist air which freezes in the upper atmosphere leading to electric load inequalities and hence lightning.

A basic understanding of convection and the emergence of thunderstorms is crucial to identify potential features that can be used in machine learning algorithms. Section 2.1 therefore gives a short introduction to the basic effects associated with thunderstorms.

2.1 The Three Phases of a Thunderstorm

The emergence of thunderstorms is usually divided into three separate phases: the developing stage, the mature stage, and finally the dissipating stage [10, 19].

Developing Stage. In the first stage of a thunderstorm emergence, warm and moist air is rising to upper levels of the atmosphere. This so called updraft can be caused by geographic reasons such as air rising at the slopes of mountains, but also by thermal reasons such as large temperature differences between the ground and the upper atmosphere, especially in summer. Given that the convection is strong enough, the warm and moist air will eventually pass the dew point where it starts to condensate, forming a so called *cumulus congestus* which is characterized by strong updrafts offering a fresh supply of warm, moist air. During this first phase, the condensed air might form first precipitation particles which however do not yet fall to the ground.

Mature Stage. In the second stage of the thunderstorm, air cools down in the troposphere, one of the upper layers of the atmosphere. If the system is strong enough, the moist air will eventually pass the point where it starts to freeze and sinks back to the ground at the sides of the updraft. This leads to a so called *downdraft*, a strong wind formed by falling (and potentially evaporating) precipitation. The *horizontal divergence* at the top of the cloud leads to the typical anvil form called *cumulonimbus*. This process of rising and finally freezing moist air is also responsible for the emergence of lightning. Cold, frozen particles may split into smaller, negatively charged particles and larger ones with a positive charge. The smaller the particle, the faster it will rise in the updraft and finally fall down back to the ground. This potentially leads to a separation of negative charge in the downdraft and positive charge in the updraft, which results in an electric load imbalance, which in turn may result in lightning which decreases this imbalance.

Dissipating Stage. In the last stage of a thunderstorm, the updrafts loose strength and finally stop. Due to the missing supply of warm, moist air, the system becomes unstable and eventually breaks down. Precipitation looses intensity, mainly consisting of the remaining condensed air at the top of the cloud.

3 RELATED WORK

Satellite-based thunderstorm forecast is usually based on the analysis of the observed atmospheric brightness temperature in different spectral channels. If the brightness temperatures or brightness temperature differences reach a critical threshold, the forecast system emits a thunderstorm warning. The satellite-based methods are often supported by NWP stability indices, which are used as an indicator for the potential energy of a system. Common indices used are the *Convective Available Potential Energy (CAPE)* [20], the *Lifted Index (LI)* [13] or the *KO Index* [7]. All these indices essentially consider the potential temperature at different levels of the atmosphere (described by pressure). The greater the differences, the more likely the atmosphere will become unstable and develop a convective system potentially leading to a thunderstorm.

Besides using such index-based forecasting methods, state-of-the-art systems such as NowCastMIX include additional data sources: Radar systems are used to detect precipitation and frozen water particles in the atmosphere. Specialized systems such as *KONRAD* [17]

and *CellMOS* [14] try to detect thunderstorm cells and predict their movement. Lightning detection systems such as *LINET* [9] measure electric waves in the atmosphere to identify lightning and their location. NWP models such as *COSMO-DE* [8] are used to identify potentially interesting areas in advance and to model the local near storm environment. A combination of these systems is then used to predict storms and follow their movement.

Although machine learning algorithms are not yet widely used in meteorology, there are first attempts to apply them to weather prediction and partially also to thunderstorm forecasting. Ruiz and Villa tried to distinguish convective and non-convective systems based on logistic regression and Random Forest models [22]. The dataset consisted of different features derived from satellite images, in particular exact values and differences for temperatures and gradients. Similar approaches have been presented by Williams et al., trying to predict convective initiation with a forecast period of one hour [25, 26] and Ahijevych et al. [6] for a forecast period of two hours. They trained Random Forests with datasets consisting of different kinds of features, covering raw satellite image values, derived fields such as CAPE or LI and radar data. Veillette et al. [24] used features based on satellite images, NWP models and environmental information to train various machine learning models on the prediction of convective initiation.

Although these approaches use similar machine learning algorithms, they differ from our approach in the feature set used for training. To the best of our knowledge, there has been no approach so far which is based on using the error of nowcasting algorithms for satellite images such as optical flow to predict thunderstorms.

4 THE ERROR IS THE FEATURE

In this section we introduce our approach to predict lighting using a model prediction error as the feature. We describe the core idea as well as the meteorological background.

4.1 Core Idea

Rather than directly deriving features from satellite images, our approach is based on the error resulting from forecasting the next image based on previous images using optical flow algorithms such as *TV-L*[1] [27]. A high-level overview of our approach is shown in Figure 1. The core idea can be formulated as follows: The movement of air within the atmosphere is a *three*-dimensional phenomenon with the third dimension depicted as brightness on satellite images. Optical flow algorithms predicting future images based on past observations however can only detect and predict *two*-dimensional movements. The error resulting from the application of optical flow might therefore be related to the vertical movement of clouds, a sign for convection potentially leading to thunderstorms. Machine Learning algorithms can then be used to learn the relation between these error values and the occurrence of lightning which is in turn a sign of thunderstorms.

Although learning from the error of a model might remind of boosting in the first place, our approach differs from such methods: Boosting algorithms iteratively train a weak classifier on a given data set and then apply a re-weighting of the samples in order to minimize the prediction error of the next weak classifier trained on the same data set. Our approach in contrast does not try to
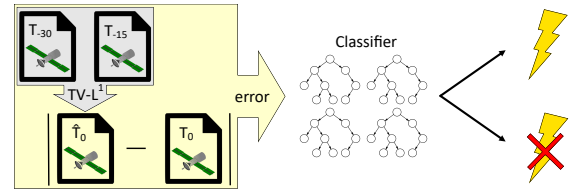

Figure 1: Overview of our approach: Two consecutive satellite images at T_{-30} and T_{-15} are read to predict the next image $\hat{T}_0$ using TV-L[1]. The error is then computed as the absolute difference between $\hat{T}_0$ and the original image T_0. We use that error to predict lightning based on different classifier.

improve the optical flow model used to predict satellite images as it would be done in boosting. Instead, we interpret the error of this first model as a feature to train a second, different model to predict lightning. The two models differ significantly in their input and output. Although both methods, boosting and our approach, learn from the error of a prediction model, we would not call our approach a boosting method, but a two stage approach instead.

4.2 Meteorological Background

The SEVIRI instrument [23] on-board of the second generation of Meteosat satellites (MSG) generates multichannel images of Europe, Africa, and the surrounding seas every 15 minutes with a sub-satellite resolution of 3×3 km for most channels. The different channels of the SEVIRI instrument are designed to measure different wave lengths. The first three channels as well as the high resolution channel essentially measure the reflection of solar light at clouds or the Earth's surface, which is closely related to the *albedo* of the repective objects. The higher the intensity of the reflected light, the brighter the tile appears on the image. As clouds have a high albedo, they appear bright in these channels. The remaining channels of the SEVIRI instrument differ from these first channels mainly in the fact that they measure primarily infrared, i.e. thermal radiation. According to Kirchhoff's law of thermal radiation, the emissivity is equal to the absorptivity for every body which absorbs and emits thermal radiation [16]. Planck's law postulates that the intensity of the emission depends on the temperature of the body: The warmer the body, the higher the intensity [21]. This relation can be used to estimate the (brightness) temperature of the surface of objects, e.g. of cloud tops, allowing us to observe changes in the (brightness) temperature using infrared satellite channels. A central feature of cloud convection is the vertical updraft of moist air which leads to an increase of the cloud albedo and cooling of the cloud, thus, to changes in the intensities. However, convective clouds move also horizontally which makes it difficult to relate changes of the observed intensities to the vertical updraft: Changes could result from real vertical updraft or from artefacts induced by differences in intensities between the Earth's surface and clouds. The central problem is the lack of three-dimensional cloud movement data as SEVIRI provides only two-dimensional images. Thus the vertical updraft can not be simply derived by the calculation of the difference in intensities between subsequent satellite images in time, preventing the detection of convection.

Optical flow algorithms such as TV-L^1 however assume that the intensity of objects does not change between consecutive images. Errors resulting from the application of optical flow on satellite images can therefore result from two reasons: The first reason is an inaccurate forecast of the movement of clouds, meaning that the actual position of a cloud object on an image was different from the predicted position. This error essentially results from weaknesses of the optical flow algorithm. Errors might however also result from cloud objects whose position was predicted correctly, but whose brightness changed between two consecutive images. As explained above, these brightness changes directly relate to the temperature of the clouds which in turn relates to their height as clouds become colder the higher they are. These (rapidly) rising clouds are however exactly the result of convection leading to thunderstorms. Detecting them would allow us to predict thunderstorms in advance, earlier than lightning detection systems or radar systems searching for precipitation particles.

5 DATA PREPROCESSING PIPELINE

As machine learning models rely on carefully engineered features in order to learn relations to some target variables, we present in this section the steps which are necessary to transform the raw data to the features used in our models.

5.1 Basic Feature Generation & Target Values

The raw input used for our approach essentially consists of two data sources: binary files containing MSG images for the different channels and CSV files containing lightning detected by the *LINET* system. As *LINET* only covers Europe, satellite data representing surrounding areas can be ignored. The first step in our pipeline was therefore a projection of the satellite images to the area covered by *LINET* using the *Satpy* library [3]. We decided to use the first nine channels of the satellite, covering a spectrum from the visible light at $0.6\mu m$ wavelength up to the infra red light at $10.8\mu m$ [23].

In the second step, the error values are computed in the following way: Two consecutive satellite images at T_{-30} and T_{-15} minutes were fed into the *TV-L^1* algorithm to compute the next image of this sequence $\hat{T}_0$. The implementation was based on the *OpenCV* python library [2], the parameters used are given in Table 7 in the appendix. In addition, we read the original image at T_0 and compute the absolute difference between these two images which we consider the error of our nowcasting. Using a classifier, we compute a target variable $\hat{Y}_{15}$ predicting the occurence of lightning within the next 15 minutes.

$$error_0 = |T_0 \; - \; \hat{T}_0| = |T_0 \; - TV\text{-}L^1(T_{-30}, T_{-15})|.$$

$$\hat{Y}_{15} = \text{classifier}(error_0).$$

The lightning data obtained from the *LINET* network was supplied as CSV files containing as attributes the time lightning occurred as well as its location, charge, and height. We transformed it to geographically tiled (or rastered) maps where each tile stores the number of lightning occurring at this location in time interval $[T; T_{+15})$ ("within the next 15 minutes"). Each tile now represents data with a temporal extension of 15 minutes and a spatial resolution equal to the one of MSG.
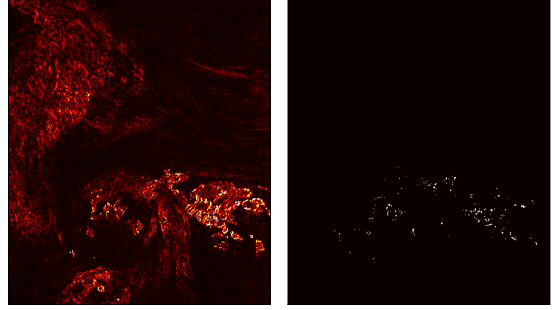

Figure 2: Example images after preprocessing: The error in channel WV6.2 at 2017-06-01 13:15 as heatmap & the lightning data for the following 15 minutes as binary mask.

For performance reasons, the images containing the error values as well as the maps containing the lightning were stored as binary numpy arrays to avoid the repetition of the costly load/parse and spatial projection operations before each training step.

5.2 Enhancing Features Using Convolution

Based on our general assumption that high error values indicate the presence of lightning, we use the absolute error values in each of the first nine channels of SEVIRI as our basic features. Figure 2 shows an optical comparison between the error in channel WV6.2 and the occurence of lightning, supporting this assumption.

As clouds and thunderstorms move over time, we also include features covering the spatial influence from nearby tiles. The use of convolution on the images containing the error values allows us to identify thunderstorms entering a tile from a neighbouring tile. We decided to explore kernels of size 3×3 up to 9×9 to cover local lightning events as well as larger areas.

For each kernel size, we explored four different convolution operations: we decided to use the maximum and minimum operations over the window specified by the kernel size to identify areas with extreme values. In addition, we explored the average value over different kernel sizes to cover information about the overall state within a certain area. Finally, we added a convolution based on the gaussian distribution to weight error values near the centre of the kernel higher than values at the border, i.e. this corresponds to a weighted average. The intuition behind this approach is that geographically close locations should influence the result more than locations at larger distances. The weights are given by the following formula where x and y denote the coordinates with respect to the centre of the kernel and σ the standard deviation:

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}.$$

Our implementation was based on the following filters defined in the *ndimage* package of the *SciPy* Python library [5]: *maximum_filter*, *minimum_filter*, *uniform_filter* and a combination of *gaussian_kernel* and *convolve* for the last operation.

This combination of four convolution kernel sizes *times* four convolution operations *times* nine channels each *plus* the raw error values of each channel yields a total of $4 * 4 * 9 + 9 = 153$ features.

We decided to train the classifiers on a per tile basis, i.e. the models were not trained on two-dimensional images, but on single

Table 1: Main characteristics of the cross-validation sets.

#	Time Range	# samples / class
0	2017-06-01 00:30 to 2017-06-08 23:00	624,877
1	2017-06-09 11:00 to 2017-06-17 09:45	65,261
2	2017-06-17 21:45 to 2017-06-25 20:15	256,456
3	2017-06-26 08:15 to 2017-07-04 06:30	291,956

tiles and their corresponding feature values. The target values were simple binary values. Each such label indicated the presence or absence of lightning at this tile given a specific offset compared to the time stamp of the error values.

6 EXPERIMENTAL SETUP

The experiments conducted in this thesis were based on 4-fold cross-validation, which allowed us to combine three sets for training while using the remaining set for testing. Our dataset approximately covers one month, from 2018-06-01 00:00 to 2018-07-04 06:30. We decided to split this set along the time axis, using a twelve hour margin between the sets. As training with highly imbalanced data is very hard, we decided to use downsampling on a per image basis, meaning that we took all tiles with a lightning event present on an image and chose the same number of tiles without lightning event at random from the same image. Due to various data errors during the extraction of the raw, binary satellite data and the pre-processing steps, we could not consider all images. The resulting cross-validation sets and the sizes of the classes are shown in Table 1. A more detailed explanation of this setup and the versions of the different Python libraries used can be found in Appendix A.

7 RESULTS

The main assumption of our approach is the correlation between high error values and the existence of lightning. A simple statistical evaluation gives evidence for this assumption: Table 2 shows the distribution of the error values in channel WV6.2 for the timespan 2018-06-01 00:30 to 2018-06-03 06:15. The first column refers to all tiles on all images within this time range while the following two columns represent the two classes, namely the *no-lightning* and the *lightning* class with respect to the next 15 minutes. Comparing these values, we can state that the mean error for the *lightning* class is higher than the one for the *no-lightning* class. Simply assuming that a high error value necessarily indicates the presence of lightning is, however, wrong as the overall maximum belongs to the *no-lightning* class. Nevertheless, we can still conclude that high values more likely indicate the presence of lightning than the absence.

Table 2 also shows the extreme imbalance of the dataset. Only 169,912 samples out of more than 230 million belong to the *lightning* class, resulting in a fraction of only 0.074% which decreases to 0.066% if computed over all available data.

7.1 Predicting the Immediate Future

To check whether our assumption is correct and we can indeed train machine learning models to predict lightning, we first conducted an experiment to forecast the immediate future: Given the error resulting from the satellite images at T_{-30}, T_{-15} and T_0, can we

Table 2: Nowcasting error distribution (2018-06-01 00:30 to 2018-06-03 06:15) in channel WV6.2, per class distribution taken with respect to lightning within the next 15 minutes.

	All tiles	No-Lightning	Lightning
# samples	230,036,544	229,866,632	169,912
mean error	0.5141	0.5128	2.3600
std. deviation	0.7680	0.7627	2.8515
min error	0.0	0.0	0.0
max error	30.8376	30.8376	30.0355

Table 3: The accuracy for each model and test set: Decision Tree (DT), Random Forest (RF), AdaBoost (AB), Gradient Boosting (GB). The last row shows the accuracy over all sets.

Test Set	DT	RF	AB	GB
0	87.494%	87.661%	90.203%	90.973%
1	86.549%	87.219%	88.648%	89.090%
2	89.946%	91.724%	90.744%	92.714%
3	84.118%	90.050%	85.483%	90.500%
overall	87.156%	89.042%	89.120%	91.123%

predict whether there will be lightning within the next 15 minutes, i.e. between T_0 and T_{+15}?

We decided to use tree-based classifiers for this first set of experiments as they provide more transparency in understanding how a prediction was made compared to neural networks. The models chosen for evaluation are simple Decision Trees as well as ensemble methods based on them: Random Forests as described by Breiman [11], Gradient Boosting as introduced by Mason et al. [18] and AdaBoost as presented by Freund and Schapire [12]. The implementations are based on the *Scikit-Learn* Python library [4]: *DecisionTreeClassifier*, *RandomForestClassifier*, *AdaBoostClassifier* and *GradientBoostingClassifier*. For most parameters we used their default values as specified in the documentation. We adopted some parameters to avoid overfitting and limit the training time. The most important parameters are depicted in Subsection B.2.

Table 3 shows the resulting accuracy values for the different models and test sets. As expected, the simple Decision Tree shows the worst performance in each test set, reaching values between 84% and 89%. Gradient Boosting performed best with accuracy results between 89% and 92%. The last row of the table shows the accuracy over all test sets, underlining a strong Gradient Boosting model, a weak Decision Tree model and the Random Forest as well as AdaBoost models in between. Considering the computational effort required to create these models, notice however that the improvement of about 2% between Random Forests and Gradient Boosting comes at the cost of a much higher training time: For the largest training set consisting of the cross-validation sets 0, 2, and 3, the training phase for Random Forest which could be conducted in parallel on all CPU cores took about 14 minutes. Gradient Boosting however which can by nature not run in parallel took about 818 minutes which is more than 54 times the duration of Random Forest.

Table 4: Main characteristics of the models over all test sets: precision (PR), recall (RE) and false positive rate (FPR).

	DT	RF	AB	GB
PR	86.22%	85.73%	89.50%	89.74%
RE	88.46%	93.68%	88.64%	92.86%
FPR	14.14%	15.59%	10.40%	10.62%

As a high overall accuracy does however not necessarily indicate an equally good performance on both classes., we also present the main characteristics resulting from the confusion matrices for all models in Table 4, created by summing up the results on the individual test sets. Especially Gradient Boosting performs equally well on both classes, leading to high precision and recall values while minimizing the false alarm rate. Random Forest does even slightly outperform Gradient Boosting in terms of recall, but at the cost of a much worse performance in terms of precision. All models except AdaBoost tend to achieve better results for the *lightning* class, which seems slightly easier to learn.

7.2 Limitations

These results indicate the potential of the method for operational usage, however, the false positive rate of 10% for the best model needs further investigation and improvement. In addition, one should keep in mind that the results shown here are based on a balanced subset of all tiles. Using the original, highly unbalanced data would not affect the values for accuracy, recall and false positive rate, but lead to a much lower precision due to the larger amount of false positives (given the assumption that the subsample is representative for the real data). The total amount of samples per class over all test sets is 1,238,550. Considering the fact that only roughly 0.066% of all tiles belong to the Lightning class and we chose the same amount of samples from the No-Lightning class, we can conclude that the overall amount of tiles in the images considered for this paper having a negative true condition is roughly in the region of 1,876,590,909. Considering the False Positive Rate of 10.62% for the Gradient Boosting model, this calculation would lead to a precision of roughly 0.57% if all tiles were considered. Applying this model alone in practice to issue warnings would therefore not be feasible. We can however calculate the false positive rate FPR necessary to achieve a better precision PR the following way:

$$PR = \frac{TP}{TP + FP} = \frac{TP}{TP + FPR * N} \Leftrightarrow FPR = \frac{(1 - PR) * TP}{PR * N}.$$

In the formula above, TP denotes the number of true positives and N the overall number of tiles with a negative true condition. However, even assuming a perfect recall, a model which aims at a precision of 20% (the value currently achieved by NowCastMIX for a one hour offset) would have to achieve a false positive rate of roughly 0.26% which seems extremely hard to achieve.

7.3 Feature Evaluation

The models considered so far all belong to the class of tree classifier algorithms which essentially all work in a very similar way: In each step, the model chooses a feature and a corresponding threshold and
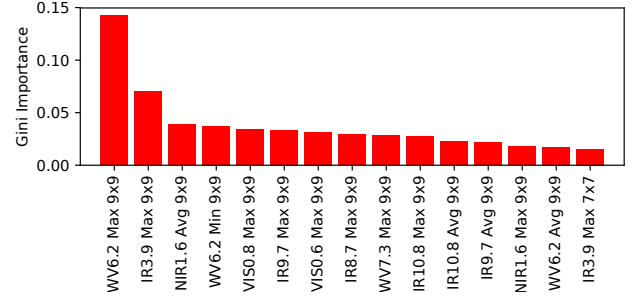

Figure 3: The top 15 features and their gini importance for the Gradient Boosting model trained on the sets 0, 2 and 3.

performs a binary split of the set of samples into two distinct subsets assigned to the left and right subtree. The decision which feature and threshold to choose is taken according to the gini impurity which measures the impurity of a node where lower values indicate purer nodes. Given the relative frequencies p_i of each class $i \in \{1, ..., J\}$, the gini impurity is defined as:

$$g(p) = 1 - \sum_{i=1}^{J} p_i^2.$$

Based on this impurity measure, we can define a feature importance measure. Given the weights of the individual trees α_t in the ensemble, the total number of samples N, the number of samples N_k, N_L, N_R reaching a parent node k as well as its left and right child node and the feature $s(k)$ used in the split of node k, we define the gini importance $FI(x_i)$ of some feature x_i as:

$$FI(x_i) = \sum_t \alpha_t \sum_{k,s(k)=x_i} \frac{N_k}{N} \left(g\left(p^k\right) - \frac{N_L}{N_k} g\left(p^L\right) - \frac{N_R}{N_k} g\left(p^R\right) \right).$$

Figure 3 shows an example plot for the top 15 features of Gradient Boosting trained on the sets 0, 2, and 3 which together form the largest training set. The maximum value within a 9×9 tile window of channel WV6.2 is the most important feature with a gini importance of about 0.14, followed by the maximum convolution with the largest kernel in channel IR3.9, however reaching just half the importance of the first feature. The third and following features keep loosing importance with values between 0.04 and 0.01. The feature importances for the other three training sets show very similar results: The maximum values of a 9×9 kernel in channels WV6.2 and IR3.9 remain the prominent features, with the first ranked feature reaching twice the gini importance of the second ranked one, followed by slightly decreasing importances for the following features. The results of Random Forest and AdaBoost models look very similar. Decision Trees rely on a much smaller subset of features: Compared to Gradient Boosting, the top ranked feature shows twice the gini importance with values of 0.35 to 0.6. The features on the following ranks show a fast drop of the importance values, indicating that the model relies on a small subset of features whereas ensemble methods such as Random Forests, AdaBoost, and Gradient Boosting obviously favour a broader range of features which potentially explains their improved accuracy.

Table 5: Channel, kernel size and convolution type usage within the top 35 features of Gradient Boosting depending on the training sets.

Feature	sets 1,2,3	sets 0,2,3	sets 0,1,3	sets 0,1,2
VIS0.6	4	4	4	4
VIS0.8	5	5	5	3
NIR1.6	2	2	2	2
IR3.9	6	8	8	8
WV6.2	4	4	4	4
WV7.3	2	2	4	3
IR8.7	3	2	0	1
IR9.7	5	5	5	7
IR10.8	4	3	3	3
1×1	0	0	0	0
3×3	0	2	3	2
5×5	2	3	4	4
7×7	8	7	8	9
9×9	25	23	20	20
Max	15	18	21	20
Min	12	9	6	8
Avg	8	8	8	7
Gaussian	0	0	0	0

Table 5 shows the number of occurrences for channels, kernel sizes as well as convolution types within the top 35 features of the Gradient Boosting model for the different training sets. Considering the channels, we can clearly state that IR3.9 belongs to the most prominent channels, appearing six to eight times among the top features, followed by IR9.7 and VIS0.8. There is however no clear concentration on a specific subset of channels: Usually all channels appear in all training sets, except for one set where IR8.7 does not appear within the top 35 features. Considering the different kernel sizes, the model clearly favours larger kernels over smaller ones. The 1×1 column which means taking the exact error value of a single tile shows no usage at all. The largest kernel size of 9×9 however is always used at least 20 times, i.e. in nearly two third of all features. A similar observation holds true for the convolution types where the maximum convolution clearly dominates with 15 to 21 appearances within the top features. In contrast, the gaussian convolution does not appear at all, indicating that this convolution type does not help the model to discriminate the samples.

8 ADDING NON-ERROR-BASED FEATURES

The experiments in Section 7 were based only on features derived from the *error* of the optical flow algorithm. What happens if we add other features? That is what this section is about.

Raw Image Values. Error values do not allow for any inference on the values of the original satellite image, making it hard to distinguish cloudy and non-cloudy tiles with similar error values. We therefore added the original values of the different channels as additional features which indeed allow for such a separation. We again use the assumption that convolution incorporating information of
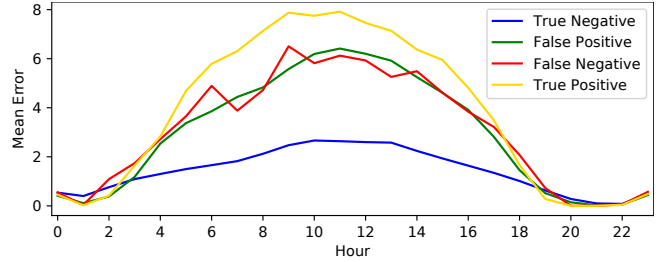

Figure 4: Mean error values for the VIS0.6 channel over time based on the results of Gradient Boosting.

surrounding tiles might be helpful and therefore apply the same convolutions as for the error values of the optical flow.

Time Of Day. The first three channels of SEVIRI cover the range of the visible light. These channels essentially show a black image over night, giving no useful information for thunderstorms occuring in the late evening or early morning. Figure 4 shows the mean values for the VIS0.6 channel over time clearly indicating very low values over night giving no useful information to distinguish the two classes. At noon however the values of this channel allow for a pretty clear distinction of lightning and no-lightning tiles. The distribution of lightning over time shows that the average number of lightning per 15 minutes stays below 400 during night and early morning before it starts to rise at approximately 10:00, reaching a peak at 15:00 with more than 1700 lightning per 15 minutes. Leaving out these channels would therefore eliminate a very useful data source for the distinction in the afternoon when most lightning occur. We therefore added an additional feature depicting the time of day as four digit number in the format [hhmm] to allow the model to take decisions based on time.

Coordinates. During sunrise, the channels covering the visible light will become brighter from north to south as the sunlight reaches tiles near the pole earlier than tiles near the equator, given the same longitude. During sunset, the same phenomenon occurs in the oppsite direction as the tiles become darker from south to north. Figure 5 shows the geographical distribution of the 10% quantile error clearly indicating that tiles in the lower half of the image remain black for a longer period of time. We therefore also include the x and y coordinate of a tile on the image as additional features to allow

Figure 5: Geographical distribution of the 10% quantile error (channel VIS0.6).

for a distinction of such cases during sunset or sunrise. Adding such geographic information might also help to distinguish mountains and plains, potentially having different triggers for thunderstorms.

Removing Features. To avoid an exploding number of features, we eliminated the gaussian convolution as it showed no improvement in the previous experiments. Considering the kernel sizes, we decided to use only two kernels, namely a 7×7 and a 13×13 kernel to incorporate the result that the previous models preferred larger kernels over smaller ones. We still keep the values for the single
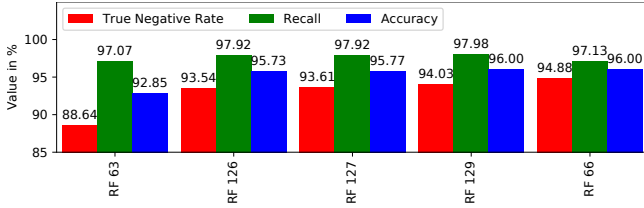
Figure 6: Random Forests with error-based features (RF63), adding raw satellite images (RF126), time of day (RF127), coordinates (RF129) or just non-error-based features (RF 66).

Offset	RF 153	RF 66	RF 129	NN 129
+0:00:00	89.04%	96.00%	96.00%	96.42%
+1:00:00	84.61%	92.09%	92.25%	92.19%
+2:00:00	80.29%	87.15%	86.30%	88.03%
+3:00:00	76.67%	82.87%	81.35%	85.48%
+4:00:00	73.68%	82.27%	80.82%	84.43%
+5:00:00	71.39%	79.99%	78.36%	83.87%

tile to allow the models to learn from differences between the value of the tile itself and convolution values which include surrounding tiles. This leads to a total of 129 features: The error values resulting from nowcasting as well as the raw satellite image values, each combined with 3 convolution types and two kernel sizes plus the features for the time of day and the coordinates.

The new feature set was fed to a Random Forest with slightly adapted parameters: We conducted a hyperparameter search with *GridSearchCV* from the Scikit-Learn library on the largest training set using three folds. We varied the number of estimators in the interval [50, 300], the maximum depth in the interval of [14, 26] and the minimum number of samples per leaf in the interval [3, 81]. A configuration with 300 estimators, a maximum depth of 26 and at least three samples per leaf yielded the best results, but started to overfit on the smaller training sets. We therefore decided to use a configuration with the maximum depth set to 18, the minimum number of samples per leaf set to nine and the number of estimators set to 200. This configuration has shown a slightly worse performance (96.00 % accuracy compared to 96.24%), but does not tend to overfit too fast on smaller training sets.

To evaluate the impact of each of the new features, we conducted several experiments, adding the new features one after the other. Figure 6 shows the resulting values for accuracy, recall and true negative rate. RF63 denotes a Random Forest just using the 63 features based on the error of the optical flow as baseline. RF126 is a Random Forest including the values for the raw satellite images, resulting in 126 features. RF127 then adds the time of day while RF129 finally uses all features including the coordinates. We also show the performance of a model using the 66 non-error-based features (RF 66) which achieves the same accuracy, however with a slightly worse detection of lightnings.

This new setup increased the accuracy up to 96.00%, clearly outperforming the Gradient Boosting model reaching only 91.12%. Considering the per class performance, this new model reaches a recall of 97.98% and a false positive rate of 5.97%, showing an equal boost for both classes compared to the previous results.

9 INCREASING THE FORECAST PERIOD

Forecasting lightning just for the next 15 minutes is probably the easiest task, but not necessarily the most useful as it gives little to no time to react. We therefore decided to conduct additional experiments to evaluate if our approach does also offer the possibility to increase the forecast period up to five hours. The basic setup remains the same, especially the data preprocessing and feature

generation steps. Instead of taking the maps which include the lightning for the next 15 minutes, we now used a specific offset to determine the maps used as target values for our machine learning models. An offset of +0:00:00 essentially just means the same as in the previous sections, namely considering the lightning for the next 15 minutes. An offset of +5:00:00 however now indicates that the model is trained to learn the presence or absence of lightning five hours in the future, i.e. between +5:00:00 and +5:15:00 compared to the last available satellite image.

Table 6 shows the results of the corresponding experiments for different models. RF 153 is the Random Forest model from the first set of experiments, using the 153 original features only considering the error values of nowcasting. RF 129 and RF 66 are the Random Forest models described in the previous section.

All the previous sections only used tree classifiers as prediction models. We decided to also test a neural network for comparison. NN 129 is a neural network based on the Keras framework [1] using 9 hidden, dense layers with sizes [64, 64, 64, 32, 32, 32, 16, 16, 16] and the leaky ReLU activation function. The single output node uses the sigmoid activation. The model was trained using the Adam optimizer and binary cross-entropy as loss function. The training time was limited to 500 epochs with early stopping if the accuracy on the test set did not improve by at least 0.01% over the last 50 epochs. The batch size was fixed to 25,000 and the feature set identical to RF 129, except for the fact that all features were normalized to the unit interval [0; 1]. We decided to not consider convolutional networks as the highly imbalanced data would most likely lead to models just predicting black images indicating no lightning at all, reaching an accuracy of more than 99.9% this way.

Considering the performance, we must state the the models lose accuracy with increasing offset. However, this result is explainable by the fact that the weather is a very chaotic phenomenon where accurate forecasts become more and more difficult with increasing forecast period. But even with the largest offset tested, the results clearly remain above the 50% margin one would expect from random guessing on a
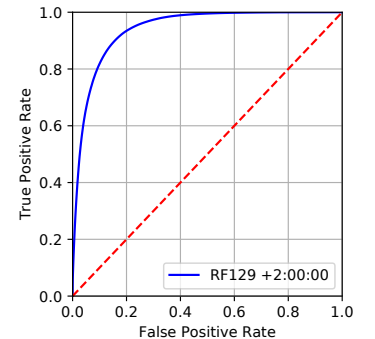

Figure 7: ROC curve for the model RF 129, offset +2:00:00.

balanced dataset. An accuracy of still more than 83% shows promising results for future work. However, a larger offset comes at the cost of an increasing number of false positives further reducing the precision of the model. This problem still needs to be solved. The receiver operating characteristic (ROC) curve of model RF 129 for a two hour offset depicted in Figure 7 however shows the capability of our model to learn the problem, reaching an area under the curve (AUC) value of 0.94. This is clearly better than the results reported by Ahijevych et al. [6] and Veillette et al. [24]. One should however keep in mind that their models were trained on predicting newly initiating convective systems which slightly differs from our approach covering all appearances of lightning. The increasing gap between the models RF 129 and NN 129 most likely relates to overfitting in RF 129 as it is weaker on smaller training sets, but stronger on larger ones. With more data available, we expect the Random Forest model to become competitive to the neural network.

10 CONCLUSION

The results of our approach seem very promising. Using just error values resulting from the nowcast of satellite images based on optical flow, different tree classifiers could be trained to predict lightning in the immediate future with an accuracy of more than 91%. Features using convolution with larger kernel sizes show the greatest impact on the accuracy of the models. The results of our models are based on a broad range of channels which are all present within the top 35 features. Adding non-error-based features improved the accuracy further, now reaching values of up to 96%. Even for the largest forecast period of five hours considered in this paper, the accuracy of the models still remain above 83% which is clearly better than the 50% which one would expect from random guessing. Comparing the results to those of Ahijevych et al. [6] or Veillette et al. [24], our approach clearly demonstrates lower false positive rates given the same recall, leading to a superior overall performance. However, the high false positive rate still needs further investigation to prepare the approach for potential future operational use.

Neural Networks as presented in the last section could offer an alternative to the tree-based models mostly used in this paper. Considering the data used as basis, we currently limit our approach to features directly related to satellite images. Including other data sources as it is done in NowCastMIX will most likely increase the performance and reduce the high false positive rate. Especially our boosted models might also benefit from a dimensionality reduction of the feature set by elimination of less important features which would allow us to grow deeper models in reasonable time, potentially yielding better results.

REFERENCES

[1] [n. d.]. Keras Deep Learning Library. https://keras.io/ Accessed: 2019-01-24.
[2] [n. d.]. OpenCV. https://opencv.org/ Accessed: 2019-01-24.
[3] [n. d.]. Satpy. https://satpy.readthedocs.io/en/latest/ Accessed: 2019-01-24.
[4] [n. d.]. Scikit-Learn. http://scikit-learn.org/stable Accessed: 2019-01-24.
[5] [n. d.]. SciPy ndimage package. https://docs.scipy.org/doc/scipy/reference/ndimage.html Accessed: 2019-01-24.
[6] David Ahijevych, James O. Pinto, John K. Williams, and Matthias Steiner. 2016. Probabilistic Forecasts of Mesoscale Convective System Initiation Using the Random Forest Data Mining Technique. *Weather and Forecasting* 31, 2 (2016), 581–599.
[7] T. Andersson, M. Andersson, C. Jacobsson, and S. Nilsson. 1989. Thermodynamic indices for forecasting thunderstorms in southern Sweden. *Meteorological Magazine* 118 (1989), 141–146.
[8] M. Baldauf, C. Gebhardt, S. Theis, B. Ritter, and C. Schraf. 2018. Beschreibung des operationellen Kürzestfristvorhersagemodells COSMO-D2 und COSMO-D2-EPS und seiner Ausgabe in die Datenbanken des DWD. (2018). https://www.dwd.de/SharedDocs/downloads/DE/modelldokumentationen/nwv/cosmo_d2/cosmo_d2_dbbeschr_version_1_0_201805.html
[9] Hans D. Betz, Kersten Schmidt, Pierre Laroche, Patrice Blanchet, Wolf P. Oettinger, Eric Defer, Z. Dziewit, and J. Konarski. 2009. LINET — An international lightning detection network in Europe. *Atmospheric Research* 91, 2 (2009), 564 – 573.
[10] Andreas Bott. 2016. *Synoptische Meteorologie: Methoden der Wetteranalyse und -prognose* (2 ed.). Springer-Verlag.
[11] Leo Breiman. 2001. Random Forests. *Machine learning* 45, 1 (2001), 5–32.
[12] Yoav Freund and Robert E Schapire. 1997. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *J. Comput. System Sci.* 55, 1 (1997), 119–139.
[13] Joseph G. Galway. 1956. The Lifted Index as a Predictor of Latent Instability. *Bulletin of the American Meteorological Society* 37, 10 (1956), 528–529.
[14] Jan Hoffmann. 2008. *Entwicklung und Anwendung von statistischen Vorhersage-Interpretationsverfahren für Gewitternowcasting und Unwetterwarnungen unter Einbeziehung von Fernerkundungsdaten.* Ph.D. Dissertation. Freie Universität Berlin. https://refubium.fu-berlin.de/handle/fub188/11705?show=full Accessed: 2019-01-24.
[15] Paul M. James, Bernhard K. Reichert, and Dirk Heizenreder. 2018. NowCastMIX: Automatic Integrated Warnings for Severe Convection on Nowcasting Time Scales at the German Weather Service. *Weather and Forecasting* 33, 5 (2018), 1413–1433.
[16] Gustav Kirchhoff. 1860. Ueber das Verhältniss zwischen dem Emissionsvermögen und dem Absorptionsvermögen der Körper für Wärme und Licht. *Annalen der Physik* 185, 2 (1860), 275–301.
[17] P Lang. 2001. Cell tracking and warning indicators derived from operational radar products. In *30th International Conference on Radar Meteorology*. American Meteorological Society, 245–247.
[18] Llew Mason, Jonathan Baxter, Peter Bartlett, and Marcus Frean. 1999. Boosting Algorithms as Gradient Descent. In *Proceedings of the 12th International Conference on Neural Information Processing Systems (NIPS'99)*. MIT Press, 512–518.
[19] H Mogil. 2007. *Extreme weather : Understanding the Science of Hurricanes, Tornadoes, Floods, Heat Waves, Snow Storms, Global Warming and Other Atmospheric Disturbances.* Black Dog & Leventhal Publishers.
[20] M. W. Moncrieff and M. J. Miller. 1976. The dynamics and simulation of tropical cumulonimbus and squall lines. *Quarterly Journal of the Royal Meteorological Society* 102, 432 (1976), 373–394.
[21] Max Karl Ernst Ludwig Planck. 1900. Zur Theorie des Gesetzes der Energieverteilung im Normalspectrum. *Verhandlungen der Deutschen Physikalischen Gesellschaft* 2 (1900), 237–245.
[22] Anne Ruiz and Nathalie Villa. 2007. Storms prediction: Logistic regression vs random forest for unbalanced data. *Case Studies in Business, Industry and Government Statistics* 1, 2 (2007), 91–101.
[23] Johannes Schmetz, Paolo Pili, Stephen Tjemkes, Dieter Just, Jochen Kerkmann, Sergio Rota, and Alain Ratier. 2002. An introduction to Meteosat Second Generation (MSG). *Bulletin of the American Meteorological Society* 83, 7 (2002), 977–992.
[24] Mark S. Veillette, Haig Iskenderian, Patrick M. Lamey, and Laura J. Bickmeier. 2013. Convective initiation forecasts through the use of machine learning methods. In *16th Conference on Aviation, Range, and Aerospace Meteorology*. American Meteorological Society.
[25] John K. Williams, David Ahijevych, Sue Dettling, and Matthias Steiner. 2008. Combining observations and model data for short-term storm forecasting. In *Remote Sensing Applications for Aviation Weather Hazard Detection and Decision Support*, Vol. 7088.
[26] John K. Williams, D. A. Ahijevych, C. J. Kessinger, T. R. Saxen, Matthias Steiner, and Susan Dettling. 2008. A machine learning approach to finding weather regimes and skillful predictor combinations for short-term storm forecasting. In *13th Conference on Aviation, Range and Aerospace Meteorology*. American Meteorological Society.
[27] Christopher Zach, Thomas Pock, and Horst Bischof. 2007. A Duality Based Approach for Realtime TV-L^1 Optical Flow. In *Proceedings of the 29th DAGM Conference on Pattern Recognition*. Springer-Verlag, 214–223.

A EXPERIMENTAL SETUP DETAILS

All Experiments were conducted on a computer equipped with two Intel Xeon E5-2600 v4 processors, 32 GB of RAM and two Nvidia GTX 1080 Ti graphics cards. The implementation was based on Python 2.7.14[1] using Satpy 0.9.2 to read and project the raw satellite data. The optical flow computation was based on OpenCV 3.4.0.12 and the models on Scikit-Learn 0.19.1.

Our training was based on a 4-fold cross-validation, meaning that the complete set of available data points was divided into four sets (folds) where three sets were combined to form the training data and the remaining set was used to test the performance of the trained model. Splitting could have been done along the geographic coordinates (meaning that we reserve a specific part of each image for testing) or along the time axis (meaning that we reserve images at a specific timestamp for testing). Splitting along the geographic coordinates would however carry the risk of overlapping training and testing sets due to the convolution operations applied during feature generation. We therefore decided to use the second approach, namely a splitting along the time axis. Our dataset approximately covers one month, from 2018-06-01 00:30 to 2018-07-04 06:30. Simply splitting this time range into training and test set at some point in time called T_S would lead to the problem that data points taken from the last image before T_S and data points taken from the first image after T_S could be highly correlated as the weather conditions did not change much during these 15 minutes. To avoid such problems, the cross-validation sets were designed such that the start and end points of the different sets are separated by a twelve hour margin. Taking into account all available tiles would have led to highly imbalanced sets as more than 99.9% of all tiles belong to the *no-lightning* class (compare Table 2). Models trained with such imbalanced data would tend to always predict the absence of lightning, achieving a trivial (and useless) accuracy of more than 99.9% this way. To avoid this problem, we decided to balance the training and test sets in the following way: Balancing through downsampling was done on a per image basis, meaning that we took all tiles with a lightning event present on an image and chose the same number of tiles without lightning event at random from the same image.

Due to various data errors during the extraction of the raw, binary satellite data and the preprocessing steps, we could not consider all images. The resulting cross-validation sets and the sizes of the classes are shown in Table 1.

B MODEL PARAMETERS

For the sake of reproducibility , we denote in this section the most relevant parameters of the different models especially if they differ from the default settings described in the documentation of the frameworks used. All parameters not mentioned in this section where left on their default values.

B.1 TV-L[1]

The implementation of TV-L[1] used in this paper was based on the DualTVL1OpticalFlow class contained in the OpenCV library. The parameters used are given in Table 7. We have not conducted a hyperparameter search for the optical flow ourselves, but used

---

[1]As the OpenCV implementation provided by DWD was based on Python 2.

Table 7: Parameters for the TV-L[1] algorithm used for optical flow computation.

Parameter	Value
tau	0.1
lambda	0.0005
theta	0.3
epsilon	0.001
outerIterations	10
innerIterations	30
gamma	0.0
nscales	7
scaleStep	0.5
warps	5
medianFiltering	1

Table 8: Parameters of the different tree classifiers. A dash means this parameter is not present for the specific model.

Parameter	DT	RF	AB	GB
Max depth	12	12	6	7
Criterion	gini	gini	gini	gini
Min # samples / leaf	0.01%	0.01%	0.01%	0.01%
# estimators	-	200	50	100
# jobs	-	16	-	-
Learning rate	-	-	1.0	0.1
Boosting criterion	-	-	-	Friedman MSE

parameters proposed by the German Meteorological Service which tested several configurations.

B.2 Tree Classifier

The tree classifier implementations were all based on the scikit-learn library using the classes *DecisionTreeClassifier*, *RandomForestClassifier*, *AdaBoostClassifier* and *GradientBoostingClassifier*. The most important parameters used for the first set of experiments with 153 features based on the error of the optical flow are depicted in Table 8. Parameters not mentioned here were left on their default values. A dash indicates that this parameter is not present for this model. For the Random Forest models described in later sections of this paper, some parameters have been adapted: The max_depth parameter was increased to 18 and the minimum samples per leaf was fixed to an absolute value of nine instead of using a percentage of the training set size.