nature methods

ARTICLES

https://doi.org/10.1038/s41592-020-01008-z

nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation

Fabian Isensee[1,2,6], Paul F. Jaeger[1,6], Simon A. A. Kohl[1,3], Jens Petersen[1,4] and Klaus H. Maier-Hein[1,5] ✉

Biomedical imaging is a driver of scientific discovery and a core component of medical care and is being stimulated by the field of deep learning. While semantic segmentation algorithms enable image analysis and quantification in many applications, the design of respective specialized solutions is non-trivial and highly dependent on dataset properties and hardware conditions. We developed nnU-Net, a deep learning-based segmentation method that automatically configures itself, including preprocessing, network architecture, training and post-processing for any new task. The key design choices in this process are modeled as a set of fixed parameters, interdependent rules and empirical decisions. Without manual intervention, nnU-Net surpasses most existing approaches, including highly specialized solutions on 23 public datasets used in international biomedical segmentation competitions. We make nnU-Net publicly available as an out-of-the-box tool, rendering state-of-the-art segmentation accessible to a broad audience by requiring neither expert knowledge nor computing resources beyond standard network training.

Semantic segmentation transforms raw biomedical image data into meaningful, spatially structured information and thus plays an essential role in scientific discovery[1,2]. At the same time, semantic segmentation is an essential ingredient in numerous clinical applications[3,4], including applications of artificial intelligence in diagnostic support systems[5,6], therapy planning support[7], intra-operative assistance[2] and tumor growth monitoring[8]. High interest in automatic segmentation methods manifests in a thriving research landscape, accounting for 70% of international image analysis competitions in the biomedical sector[9].

Despite the recent success of deep learning-based segmentation methods, their applicability to specific image analysis problems of end-users is often limited. The task-specific design and configuration of a method requires high levels of expertise and experience, with small errors leading to large drops in performance[10]. Especially in three-dimensional (3D) biomedical imaging, for which dataset properties such as imaging modality, image size, (anisotropic) voxel spacing and class ratio vary drastically, this process can be cumbersome and successful configurations from one dataset rarely translate to another. The numerous expert decisions involved in adapting and training a neural network range from exact network architecture to training schedule and methods for data augmentation or post-processing. Each of the interdependent sub-components is controlled by essential parameters such as learning rate, batch size or class sampling strategy[10]. An additional layer of complexity in the overall setup is posed by the hardware available for training and inference[11]. Purely empirical optimization of the co-dependent design choices in this high-dimensional space, as proposed by previous studies in the field of automatic machine learning (AutoML)[12], amplifies both the number of required training cases, as well as compute resources, by orders of magnitude and typically covers a mere fraction of the segmentation pipeline (such as the architecture or the data augmentation), leaving a considerable proportion of its configuration to the experimenter[12]. Moreover, the application of AutoML to new datasets comes with its own set of required expert choices, for example, when considering the construction of a sensible problem-specific search space[12]. As our analysis of the current landscape in international biomedical segmentation challenges indicates (Results), these practical limitations commonly leave users with a manual and an iterative trial-and-error process during method design that is mostly driven by individual experience, is only scarcely documented and often results in suboptimal segmentation pipelines[10,13].

In this work, we outline a new path between the status quo of primarily expert-driven method configuration in biomedical segmentation on one side and primarily data-driven AutoML approaches on the other. Specifically, we define a recipe hereafter that systematizes the configuration process on a task-agnostic level and drastically reduces the search space for empirical design choices when given a new task.

1. Collect design decisions that do not require adaptation between datasets and identify a robust common configuration ('fixed parameters').
2. For as many of the remaining decisions as possible, formulate explicit dependencies between specific dataset properties ('dataset fingerprint') and design choices ('pipeline fingerprint') in the form of heuristic rules to allow for almost-instant adaptation on application ('rule-based parameters').
3. Learn only the remaining decisions empirically from the data ('empirical parameters').

Our implementation of this recipe was developed and validated on the ten datasets provided by the Medical Segmentation Decathlon. The resulting segmentation method, which we call nnU-Net, is able to perform automated configuration for arbitrary new datasets. In contrast to existing research methods, nnU-Net is holistic, that is, its automated configuration covers the entire segmentation pipeline (including essential topological parameters of the network architecture) without any manual decisions. Moreover, automated configuration in nnU-Net is fast, comprising a simple execution of rules and only a few empirical choices to be made, thus

[1]Division of Medical Image Computing, German Cancer Research Center, Heidelberg, Germany. [2]Faculty of Biosciences, University of Heidelberg, Heidelberg, Germany. [3]DeepMind, London, UK. [4]Faculty of Physics & Astronomy, University of Heidelberg, Heidelberg, Germany. [5]Pattern Analysis and Learning Group, Department of Radiation Oncology, Heidelberg University Hospital, Heidelberg, Germany. [6]These authors contributed equally: Fabian Isensee and Paul F. Jaeger. ✉e-mail: k.maier-hein@dkfz.de

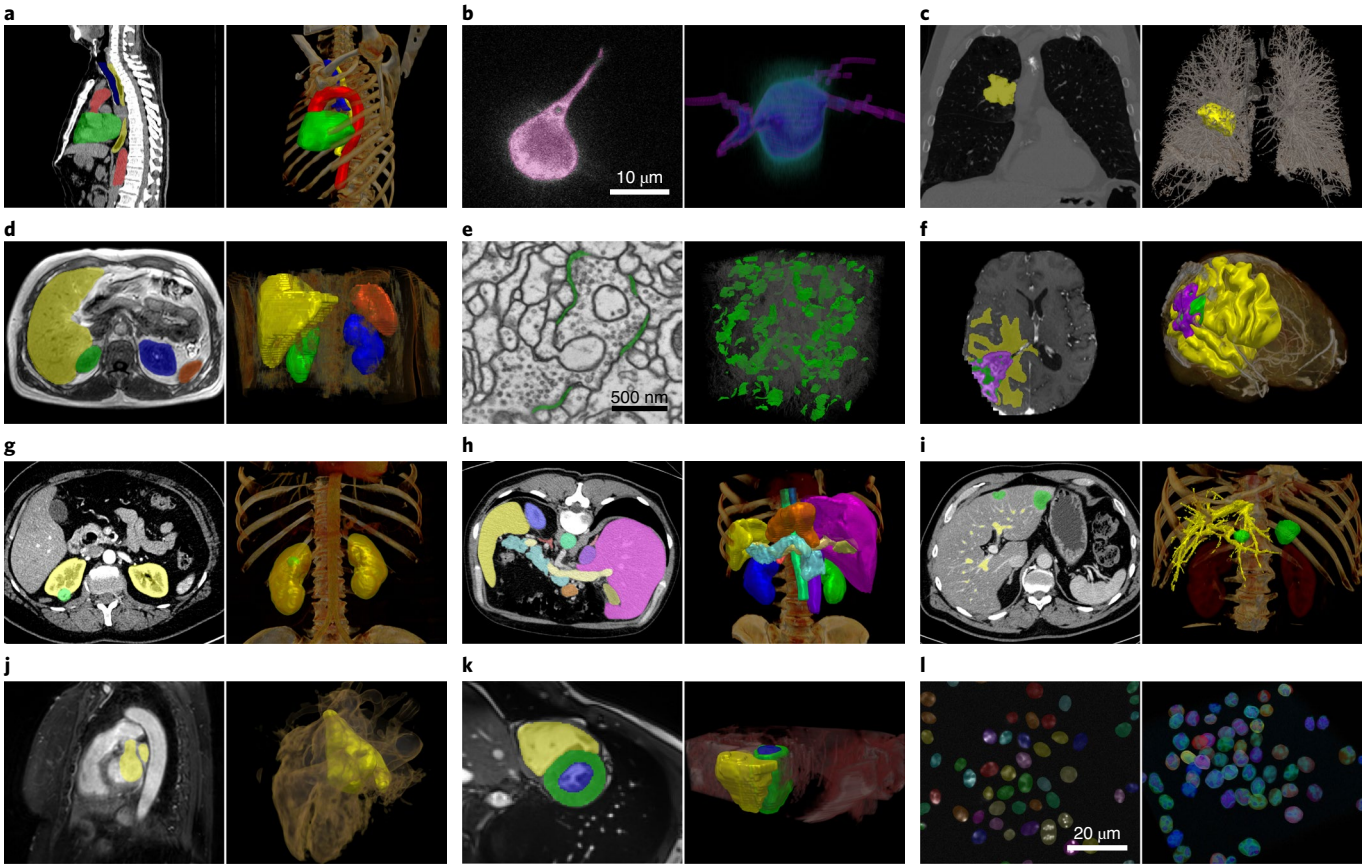


Fig. 1 | nnU-Net handles a broad variety of datasets and target image properties. All examples originate from test sets of different international segmentation challenges to which nnU-Net was applied. Target structures for each dataset are shown in 2D projected onto the raw data (left) and in 3D together with a volume rendering of the raw data (right). All visualizations were created with the MITK Workbench[35]. **a**, Heart (green), aorta (red), trachea (blue) and esophagus (yellow) in CT images (dataset (D)18)[14]. **b**, A549 lung cancer cells (purple) in FM (D22)[36,37]. **c**, Lung nodules (yellow) in CT images (D6)[14]. **d**, Liver (yellow), spleen (orange), left and right kidneys (blue and green, respectively) in T1 in-phase MRI (D16)[20]. **e**, Synaptic clefts (green) in EM scans (D19) (https://cremi.org/). **f**, Edema (yellow), enhancing tumor (purple), necrosis (green) in MRI (T1, T1 with contrast agent, T2, FLAIR) (D1)[14,38]. **g**, Kidneys (yellow) and kidney tumors (green) in CT images (D17)[21]. **h**, Thirteen abdominal organs in CT images (D11)[16]. **i**, Hepatic vessels (yellow) and liver tumors (green) in CT (D8)[14]. **j**, Left ventricle (yellow) in MRI (D2)[14]. **k**, Right ventricle (yellow), left ventricular cavity (blue) and myocardium of left ventricle (green) in cine MRI (D13)[6]. **l**, HL60 cell nuclei (instance segmentation, one color per instance) in FM (D21)[39].

requiring virtually no compute resources beyond standard model training. Lastly, nnU-Net is data efficient; encoding design choices based on a large and diverse data pool serves as a strong inductive bias for application to datasets with limited training data.

The general applicability of nnU-Net's automated configuration is demonstrated in 13 additional datasets. Altogether, we report results on 53 segmentation tasks, covering an unprecedented diversity of target structures, image types and image properties. As an open source tool, nnU-Net can simply be trained out-of-the box to generate state-of-the-art segmentations.

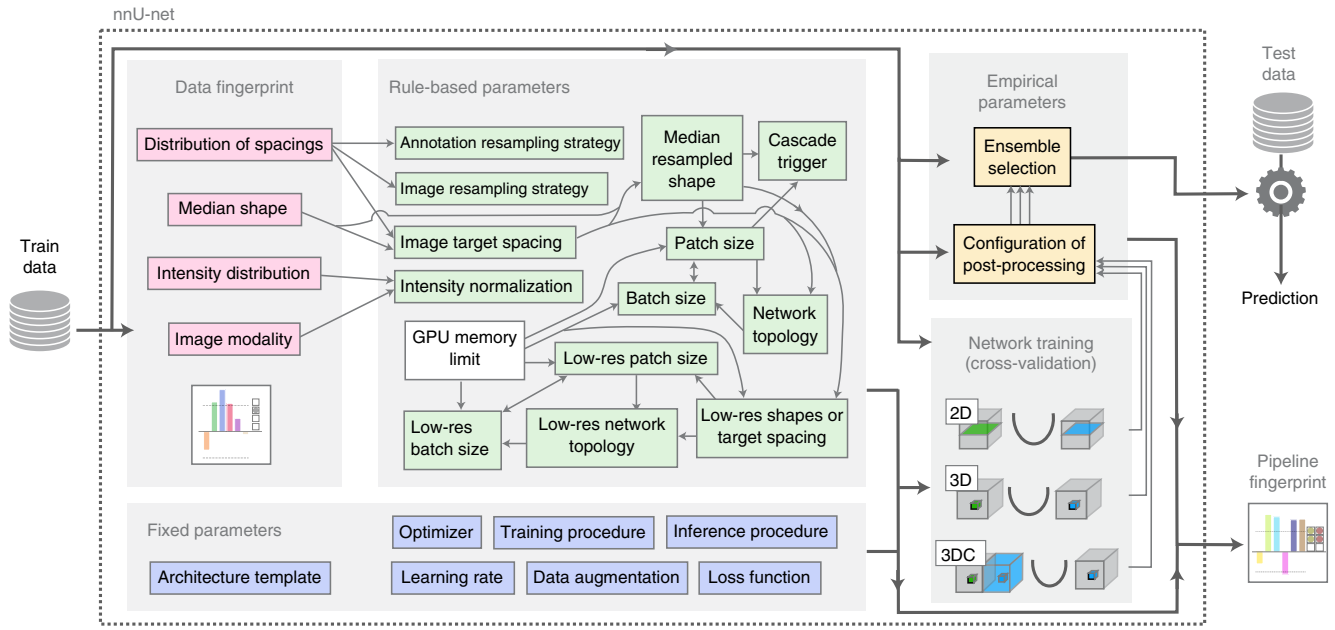
Results

nnU-Net is a deep learning-based segmentation method that automatically configures itself, including preprocessing, network architecture, training and post-processing for any new task in the biomedical domain. Exemplary segmentation results generated by nnU-Net for a variety of datasets are shown in Fig. 1.

nnU-Net automatically adapts to any new dataset. Fig. 2 shows how nnU-Net systematically addresses the configuration of entire segmentation pipelines and provides a visualization and description of the most relevant design choices.

nnU-Net development. The automatic configuration of nnU-Net is based on distilling domain knowledge into three parameter groups: fixed, rule-based and empirical parameters. First, we collect all design choices that do not require adaptation between datasets (such as setting the architecture template to 'U-Net-like') and optimize their joint configuration for robust generalization on our development datasets. Second, for as many of the remaining decisions as possible, we formulate explicit dependencies between the 'dataset fingerprint', a standardized dataset representation comprising key properties such as image size, voxel spacing information or class ratios, and the 'pipeline fingerprint', which we define as the entirety of choices being made during method design. The dependencies are modeled in the form of interdependent heuristic rules that allow for almost-instant execution upon application. To give an example for such rules, the interdependent configuration of batch size, patch size and network topology is based on the following three principles.

- Larger batch sizes allow for more accurate gradient estimates and are thus preferable (up to a sweet spot typically not reached in our domain), but in practice any batch size larger than one already results in robust training.



Design choice	Required input	Automated (fixed, rule-based or empirical) configuration derived by distilling expert knowledge (more details in online methods)
Learning rate	–	Poly learning rate schedule (initial, 0.01)
Loss function	–	Dice and cross-entropy
Architecture template	–	Encoder–decoder with skip-connection ('U-Net-like') and instance normalization, leaky ReLU, deep supervision (topology-adapted in inferred parameters)
Optimizer	–	SGD with Nesterov momentum ($\mu = 0.99$)
Data augmentation	–	Rotations, scaling, Gaussian noise, Gaussian blur, brightness, contrast, simulation of low resolution, gamma correction and mirroring
Training procedure	–	1,000 epochs × 250 minibatches, foreground oversampling
Inference procedure	–	Sliding window with half-patch size overlap, Gaussian patch center weighting
Intensity normalization	Modality, intensity distribution	If CT, global dataset percentile clipping & z score with global foreground mean and s.d. Otherwise, z score with per image mean and s.d.
Image resampling strategy	Distribution of spacings	If anisotropic, in-plane with third-order spline, out-of-plane with nearest neighbor Otherwise, third-order spline
Annotation resampling strategy	Distribution of spacings	Convert to one-hot encoding → If anisotropic, in-plane with linear interpolation, out-of-plane with nearest neighbor Otherwise, linear interpolation

Image target spacing	Distribution of spacings	If anisotropic, lowest resolution axis tenth percentile, other axes median. Otherwise, median spacing for each axis. (computed based on spacings found in training cases)
Network topology, patch size, batch size	Median resampled shape, target spacing, GPU memory limit	Initialize the patch size to median image shape and iteratively reduce it while adapting the network topology accordingly until the network can be trained with a batch size of at least 2 given GPU memory constraints. for details see online methods.
Trigger of 3D U-Net cascade	Median resampled image size, patch size	Yes, if patch size of the 3D full resolution U-Net covers less than 12.5% of the median resampled image shape
Configuration of low-resolution 3D U-Net	Low-res target spacing or image shapes, GPU memory limit	Iteratively increase target spacing while reconfiguring patch size, network topology and batch size (as described above) until the configured patch size covers 25% of the median image shape. For details, see online methods.
Configuration of post-processing	Full set of training data and annotations	Treating all foreground classes as one; does all-but-largest-component-suppression increase cross-validation performance? Yes, apply; reiterate for individual classes No, do not apply; reiterate for individual foreground classes
Ensemble selection	Full set of training data and annotations	From 2D U-Net, 3D U-Net or 3D cascade, choose the best model (or combination of two) according to cross-validation performance

Fig. 2 | Proposed automated method configuration for deep learning-based biomedical image segmentation. Given a new segmentation task, dataset properties are extracted in the form of a 'dataset fingerprint' (pink). A set of heuristic rules models parameter interdependencies (shown as thin arrows) and operates on this fingerprint to infer the data-dependent 'rule-based parameters' (green) of the pipeline. These are complemented by 'fixed parameters' (blue), which are predefined and do not require adaptation. Up to three configurations are trained in a five-fold cross-validation. Finally, nnU-Net automatically performs empirical selection of the optimal ensemble of these models and determines whether post-processing is required ('empirical parameters', yellow). The table on the bottom shows explicit values as well as summarized rule formulations of all configured parameters. Res., resolution.

- Larger patch size during training increases the contextual information absorbed by the network and is thus crucial for performance.
- The topology of a network should be deep enough to guarantee an effective receptive field size at least as large as the patch size, so that contextual information is not discarded.

Distilling this knowledge into successful method design results in the following heuristic rule: "initialize the patch size to median image shape and iteratively reduce it while adapting the network topology accordingly (including network depth, number and position of pooling operations along each axis, feature map sizes and convolutional kernel sizes) until the network can be trained with a batch size of at least two given GPU memory constraints." A detailed description of all heuristic rules is provided in the online methods, and a compilation of guiding principles used for the derivation of rules is provided in Supplementary Note 2. Third, we set only the remaining design choices, that is, model selection and post-processing, to be decided empirically based on the training data during application. Our implementation of this recipe, which we call nnU-Net, was exclusively developed on a set of ten development datasets originating from the Medical Decathlon Segmentation Challenge[14].

nnU-Net application. When applying nnU-Net to a new dataset, the automatic configuration of nnU-Net runs without manual intervention. Thus, with the exception of the few remaining empirical choices to be made, no additional computational cost beyond a standard network training procedure is required. nnU-Net's automated method configuration starts with the extraction of the dataset fingerprint and subsequent execution of heuristic rules. By default, nnU-Net generates three different U-Net[15] configurations: a two-dimensional (2D) U-Net, a 3D U-Net that operates at full image resolution and a 3D U-Net cascade in which the first U-Net operates on downsampled images, and the second is trained to refine the segmentation maps created by the former at full resolution. After cross-validation, nnU-Net empirically chooses the best performing configuration or ensemble. Finally, nnU-Net empirically opts for 'non-largest component suppression' as a post-processing step if performance gains are measured. The output of nnU-Net's automated configuration and training process are fully trained models that can be deployed to make predictions on unseen images. We demonstrate the generalization ability of the design choices encoded in nnU-Net's fixed, rule-based and empirical parameters by applying it to 13 additional datasets.

An in-depth description of the methodology behind nnU-Net, as well as its overarching design principles, is provided in the Methods and Supplementary Note 2, respectively. Segmentation pipelines generated by nnU-Net for all datasets are provided in Supplementary Note 6.

nnU-Net handles a wide variety of target structures and image properties. We demonstrate the value of nnU-Net as an out-of-the-box segmentation tool by applying it to 11 international biomedical image segmentation challenges comprising 23 different datasets and 53 segmentation tasks[6,14,16–24] (https://cremi.org/). This selection comprised a variety of organs, organ substructures, tumors, lesions and cellular structures in 2D as well as 3D images acquired by magnetic resonance imaging (MRI), computed tomography (CT), electron microscopy (EM) and fluorescence microscopy (FM). 'Challenges' are international competitions that aim to assess the performance of multiple algorithms in a standardized environment[9]. In all segmentation tasks, nnU-Net was trained from scratch using only the provided challenge data. Qualitatively, we observed that nnU-Net could handle a large disparity in dataset properties and diversity in target structures; that is, generated pipeline configurations were in line with what human experts consider a reasonable or sensible setting (Supplementary Note 3, sections 1 and 2). Examples for segmentation results generated by nnU-Net are presented in Fig. 1.

nnU-Net outperforms specialized pipelines in a range of diverse tasks. Fig. 3 provides an overview of the quantitative results achieved by nnU-Net and the competing challenge teams across all 53 segmentation tasks. Despite its generic nature, nnU-Net outperforms most existing segmentation solutions, even though the latter were specifically optimized for the respective task. Overall, nnU-Net sets a new state of the art in 33 of 53 target structures and otherwise shows performances on par with or close to the top leaderboard entries.

Details in method configuration have more impact on performance than do architectural variations. To provide a more profound picture of the current practice in deep learning-based biomedical image segmentation, we analyze as examples the participating algorithms in the recent Kidney and Kidney Tumor Segmentation (KiTS) 2019 challenge hosted by the Medical Image Computing and Computer Assisted Intervention (MICCAI) society[25]. The MICCAI society has consistently hosted at least 50% of all annual biomedical image analysis challenges[9]. With more than 100 competitors, the KiTS challenge was the largest competition at MICCAI 2019. The first observation was that AutoML approaches were noticeably absent from the leaderboard. Only one submission (rank 18 of 100) reported the selection of "a few hyperparameters via grid search" (http://results.kits-challenge.org/miccai2019/manuscripts/peekaboo_2.pdf), while manual trial-and-error optimization represented the undeniable status quo. Notably, this observation was not specific to KiTS; we are not aware of successful submissions employing AutoML in any biomedical image segmentation competition. Fig. 4a provides an overall summary of the KiTS leaderboard (http://results.kits-challenge.org/miccai2019), revealing further insights on the current landscape of deep learning-based segmentation method design. First, the top 15 methods were derived from the (3D) U-Net architecture from 2016 (refs. [15,26]), confirming its impact on the field of biomedical image segmentation. Second, contributions using the same type of network resulted in performances spread across the entire leaderboard. Third, in examining the top 15 methods, none of the commonly used architectural modifications (for example, residual connections[27,28], dense connections[29,30], attention mechanisms[31] or dilated convolutions[32,33]) represented a necessary condition for good performance on the KiTS task.

Fig. 4b underscores the importance of finding good method configurations. It illustrates an analysis of algorithms that all used the same architecture variant as the challenge-winning contribution, a 3D U-Net with residual connections. While one of these methods won the challenge, other contributions based on the same principle cover the entire range of evaluation scores and rankings. Key configuration parameters were selected from respective pipeline fingerprints, illustrating the co-dependent design choices that each team made during method configuration. The drastically varying configurations submitted by contestants indicate the underlying complexity of the high-dimensional optimization problem that is implicitly posed by configuring a deep learning method for biomedical image segmentation.

nnU-Net experimentally underscores the relative importance of method configuration over architectural variations in the KiTS dataset by setting a new state of the art on the open leaderboard (nnU-Net was submitted to the leaderboard after the original challenge ended and was thus not part of the original leaderboard analysis. The methods analyzed in Fig. 4 are also listed in the open leaderboard.) with a plain 3D U-Net architecture. This observation is in line with our results from 22 additional datasets (Fig. 3).

Different datasets require different pipeline configurations. We extracted the data fingerprints of 23 biomedical segmentation datasets. As displayed in Fig. 5, this documented an exceptional dataset diversity in biomedical imaging, and revealed the fundamental reason behind the lack of out-of-the-box segmentation algorithms: the complexity of method configuration is amplified by the fact that

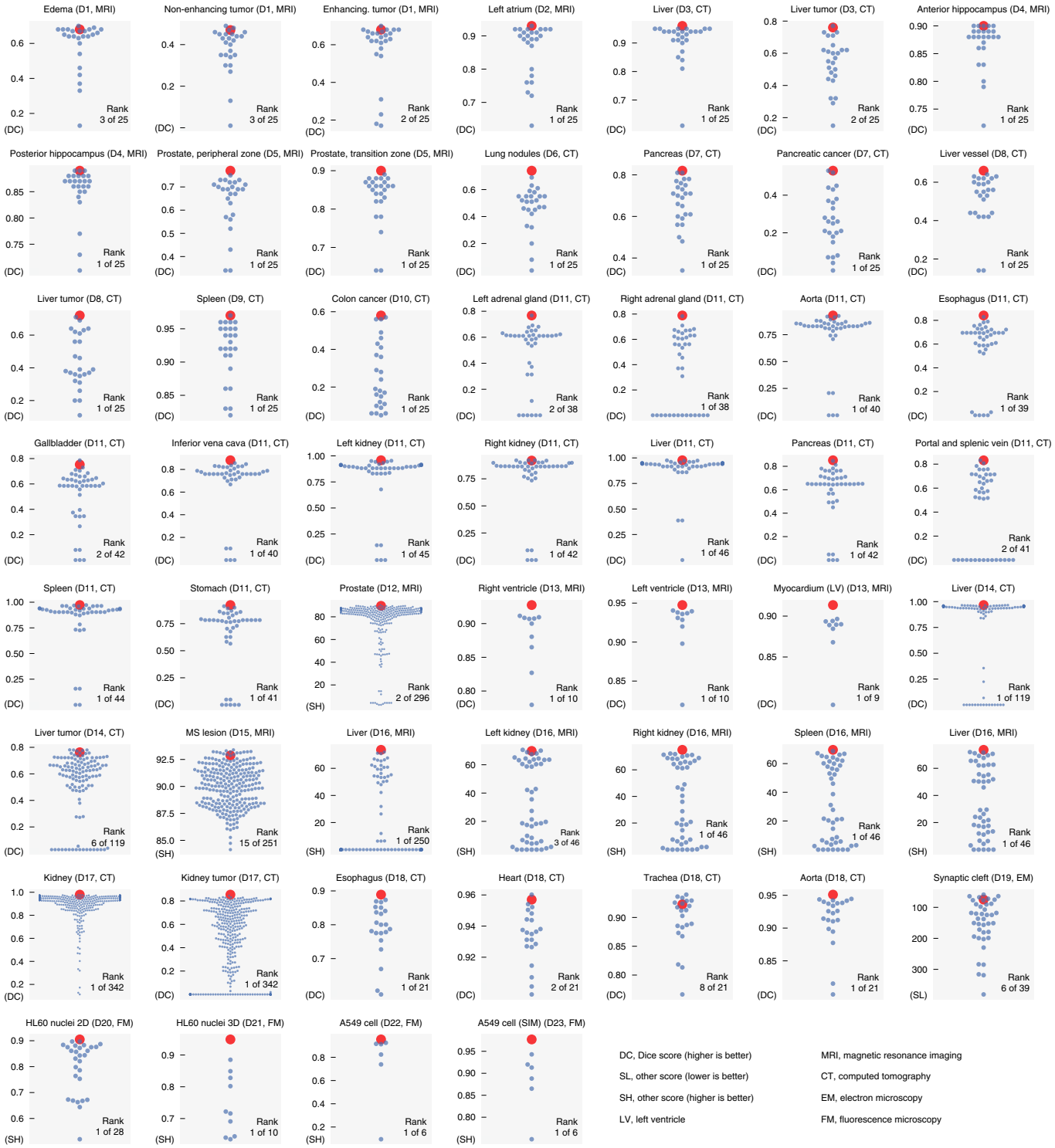


Fig. 3 | nnU-Net outperforms most specialized deep learning pipelines. Quantitative results from all international challenges that nnU-Net competed in. For each segmentation task, results achieved by nnU-Net are highlighted in red; competing teams are shown in blue. For each segmentation task, nnU-Net's rank as well as the total number of competing algorithms is displayed in the bottom-right corner of each plot. Note that for the CHAOS challenge (D16), we only participated in two of the five subtasks for reasons outlined in section 9 of Supplementary Note 6. The Cell Tracking Challenge leaderboard (D20–D23) was last accessed on 30 July 2020; all remaining leaderboards were last accessed on 12 December 2019. SIM, simulated.

suitable pipeline settings either directly or indirectly depend on the data fingerprint under potentially complex relations. As a consequence, pipeline settings that are identified as optimal for one dataset (such as KiTS, see above) may not generalize to others, resulting in a need for reoptimization for each individual dataset. nnU-Net addresses this challenge by identifying robust design decisions and explicitly modeling key interdependencies (Fig. 2).

Multiple tasks enable robust design decisions. nnU-Net's automatic method configuration can be exploited by researchers for the

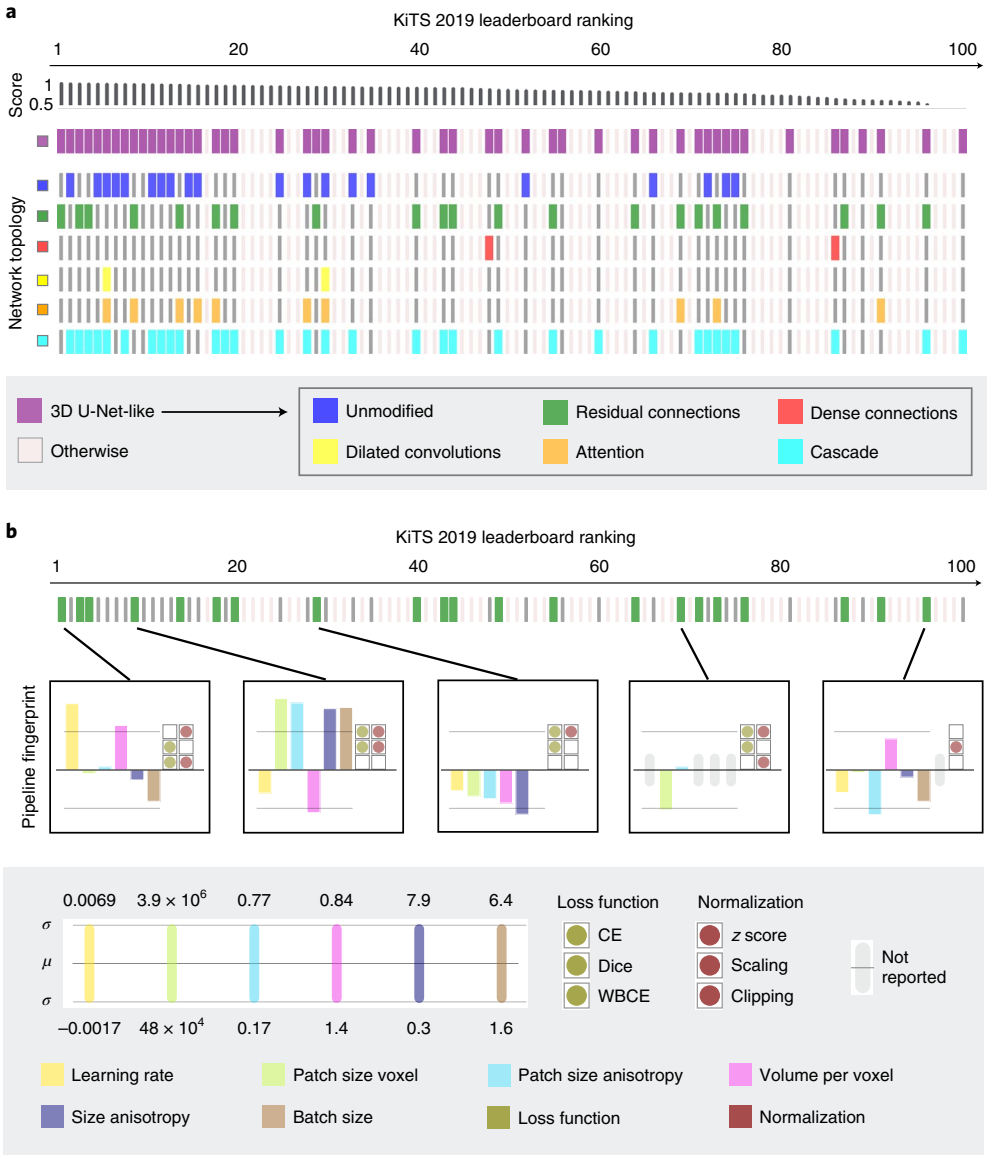


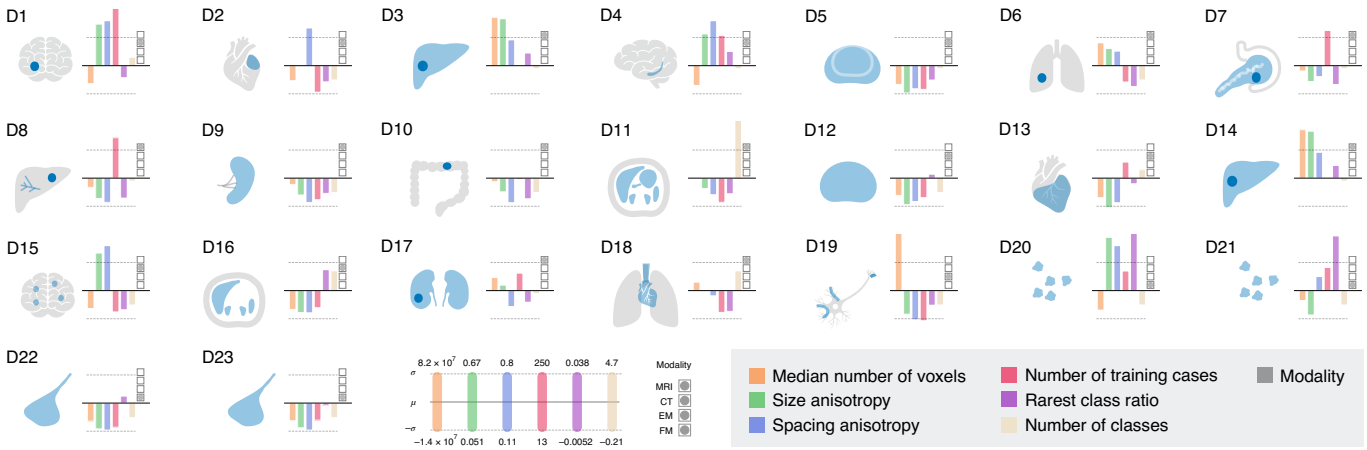
Fig. 4 | Pipeline fingerprints from KiTS 2019 leaderboard entries. a, Coarse categorization of leaderboard entries by architecture variation. All top 15 contributions were encoder–decoder architectures with skip-connections, 3D convolutions and output stride 1 ('3D U-Net-like', purple). **b**, Finer-grained key parameters selected from the pipeline fingerprints of all non-cascaded 3D U-Net-like architectures with residual connections (displayed on a z-score normalized scale). Abbreviations, CE, cross-entropy loss function; Dice, soft Dice loss function; WBCE, weighted binary CE. Further information on the KiTS 2019 challenge is available at http://results.kits-challenge.org/.

development of new segmentation methods. Novel ideas can easily be integrated into nnU-Net and thus tested across multiple datasets without having to manually reconfigure the entire pipeline for each dataset. To demonstrate the benefits of this approach, and also to support some of the core design choices made in nnU-Net, we systematically tested the performance of common pipeline variations by systematically modifying some of nnU-Net's fixed parameters. The following variations were evaluated across ten different datasets and were compared against our default nnU-Net configuration, which served as a baseline in these experiments (Fig. 6).

The volatility of the ranking between datasets demonstrates how single design choices can affect segmentation performance depending on the dataset. The results clearly show that caution is required when drawing methodological conclusions from evaluations that are based on an insufficient number of datasets. While five of the nine variants achieved rank 1 in at least one of the datasets, none of them exhibited consistent improvements across the ten tasks. The original nnU-Net configuration showed the best generalization and ranked first when results from all datasets were aggregated.

Discussion

We present nnU-Net, a deep learning-based segmentation method that automatically configures itself including preprocessing, network architecture, training and post-processing for any new task in the biomedical domain. nnU-Net sets a new state of the art for the majority of tasks on which it was evaluated, outperforming all respective specialized processing pipelines. The strong performance of nnU-Net is not achieved by a new network architecture, loss function or training scheme (hence the name nnU-Net, 'no new net'), but by systematizing the complex process of manual method configuration, which was previously addressed either by cumbersome manual tuning or purely empirical approaches with practical limitations. We hypothesize that the reason behind nnU-Net's state-of-the-art performance lies in the distillation of knowledge from a large pool



D1 MSD, brain tumor (edema, necrosis, enhancing tumor)
D2 MSD, heart (left atrium)
D3 MSD, liver (liver, liver tumor)
D4 MSD, hippocampus (anterior and posterior)
D5 MSD, prostate (peripheral zone, transition zone)
D6 MSD, lung (lung nodules)
D7 MSD, pancreas (pancreas, pancreatic tumor)
D8 MSD, hepatic vessel (hepatic vessels, liver tumors)
D9 MSD, spleen (spleen)
D10 MSD, colon (colon cancer)
D11 BCV-Abdomen (13 abdominal organs)
D12 PROMISE12 (prostate)
D13 ACDC (left and right ventricle, myocardium)
D14 LiTS (liver, liver tumor)
D15 MSLes (MS lesions)
D16 CHAOS (liver, spleen, left and right kidneys)
D17 KiTS (kidneys, kidney tumor)
D18 SegTHOR (heart, aorta, esophagus, trachea)
D19 CREMI (synaptic cleft)
D20 CTC - Fluo-N2DH-SIM+ (HL60 nuclei)
D21 CTC - Fluo-N3DH-SIM+ (HL60 nuclei)
D22 CTC - Fluo-C3DH-A549 (A549 cell)
D23 CTC - Fluo-C3DH-A549-SIM (A549 cell)

MSD, Medical Segmentation Decathlon; CTC, Cell Tracking Challenge

Fig. 5 | Data fingerprints across different challenge datasets. Data fingerprints show the key properties (displayed with *z*-score normalization over all datasets on a scale of one s.d. around the mean) for the 23 datasets used in the nnU-Net experiments (see Supplementary Note 1 for detailed dataset descriptions).

of datasets into a set of robust design choices that translates into powerful inductive biases when applied to a new dataset, and this allows for generalization abilities beyond that of models configured on a single dataset. Moreover, by condensing domain knowledge into a set of fixed, rule-based and empirical parameters, we outline a new path for automated method configuration that is computationally feasible, while at the same time covers the entire segmentation pipeline, including essential topological parameters of the network architecture. nnU-Net is a new segmentation tool that can be applied out-of-the-box, without requiring any user intervention, to a large range of biomedical imaging datasets and is thus ideal for users who require access to semantic segmentation methods and do not have the expertise, time, data or computing resources required to adapt existing solutions to their problem.

Our analysis of the KiTS leaderboard reveals a manual and not sufficiently systematic status quo of method configuration in biomedical image segmentation and highlights several implications for current research in the field. For instance, we observed that contributions using an identical type of network resulted in performances spread across the entire leaderboard (Fig. 4). This observation is in line with Litjens et al., who, in their review found that "many researchers use the exact same architectures" "but have widely varying results" (ref. [10]). There are several possible reasons for why performance improvements based on architectural extensions proposed by the literature may not translate to all datasets in the domain. First, the diversity of datasets in the biomedical domain requires dedicated method configurations (Fig. 5). As a consequence, the quality of method configuration on a new dataset may overshadow the effect of the evaluated architectural modifications. This explanation is in line with an observation by Litjens et al., who concluded that "the exact architecture is not the most important determinant in getting a good solution" (ref. [10]) and is supported by nnU-Net's state-of-the-art results based on a strong method configuration in combination with a simple U-Net architecture. Second, in current research practice, evaluation is rarely performed on more than two datasets, and, even then, the datasets come with largely overlapping properties (such as both being abdominal CT scans). As demonstrated in our multi-dataset study (Fig. 6), such evaluation is unsuitable for drawing general methodological conclusions. We relate the lack of sufficiently broad evaluations to the substantial effort required when manually adapting the configuration of proposed methods as well as existing pipelines (that is, baselines) to individual datasets. Crucially, this cumbersome process can also result in suboptimally configured baselines, causing a potential bias in literature. nnU-Net is able to alleviate these bottlenecks of current research. On the one hand, nnU-Net represents a new method that does not require manual task-specific adaptation and as such can readily serve as a strong and standardized baseline on any new segmentation task. On the other hand, nnU-Net can help to increase the number of datasets used for evaluation in the field by serving as a scalable experiment framework in which researchers easily implement methodological modifications.

While nnU-Net was shown to robustly find high quality configurations on new datasets, task-specific empirical optimization may have the potential to further improve the segmentation performance. However, as elaborated in the introduction, practical limitations of empirical AutoML approaches currently hamper their application to biomedical image segmentation. Another limitation is the lack of transparency associated with data-driven optimization ('black box algorithms' (ref. [12])) as compared to nnU-Net, for which, due to the underlying use of guiding principles, each design decision can be traced back either to certain dataset properties or a limited set of empirical experiments. Looking forward, we consider our work complementary to empirical AutoML research; nnU-Net may serve as a basis for holistic automation that can be enhanced by empirical optimization of selected design decisions such as data augmentation or network architecture.

Despite its strong performance across 53 diverse tasks, there might be segmentation tasks for which nnU-Net's automatic adaptation is suboptimal. For example, nnU-Net was developed with a focus on the Dice coefficient as performance metric. Some tasks, however, might require highly domain-specific target metrics for

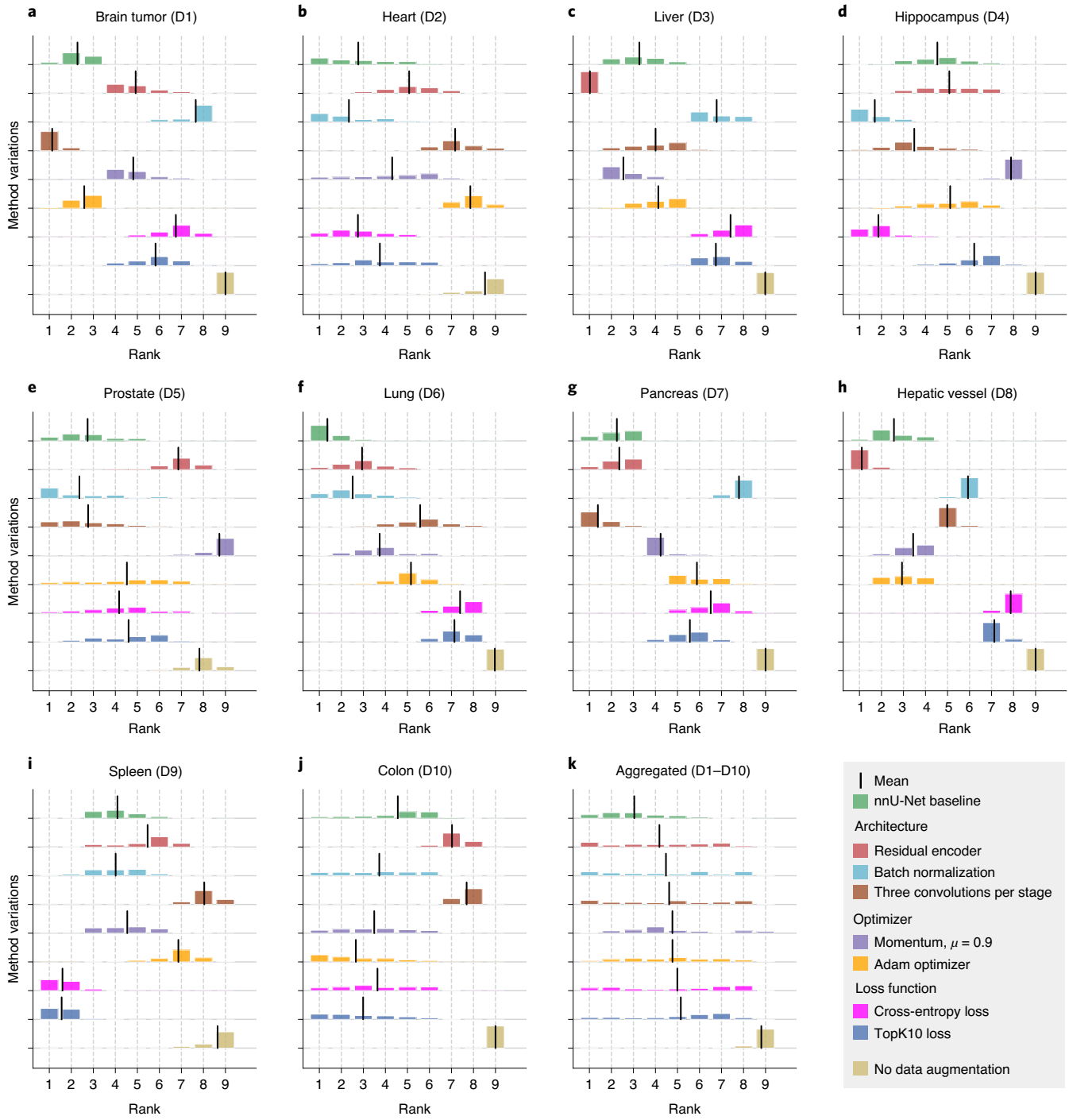


Fig. 6 | Evaluation of design decisions across multiple tasks. a–j, Evaluation of exemplary model variations on ten datasets from the medical segmentation decathlon (D1–D10, see Fig. 5 for dataset references): the application of two alternative loss functions (cross-entropy and TopK10[40]), the introduction of residual connections in the encoder[41], using three instead of two convolutions per resolution (resulting in a deeper network architecture), two modifications of the optimizer (a reduced momentum term and an alternative optimizer (Adam[42])), batch normalization[43] instead of instance normalization[44] and the omission of data augmentation (ablation experiments for further design choices can be found in Supplementary Note 8). Ranking stability was estimated for every dataset by aggregating validation splits of the five-fold cross-validation into one large validation set. One thousand virtual validation sets were generated via bootstrapping (drawn with replacement). Algorithms were ranked on each virtual validation set, resulting in a distribution over rankings as suggested by the challengeR tool[45]. **k**, The aggregation of rankings across datasets yields insights into which design decisions robustly generalize.

evaluation, which could influence method design. Also, dataset properties that are yet unconsidered could exist, which may cause suboptimal segmentation performance. One example is the synaptic cleft segmentation task of the CREMI challenge (https://cremi.org). While nnU-Net's performance is highly competitive (rank 6 of 39), manual adaptation of the loss function, as well as EM-specific preprocessing, may be necessary to surpass state-of-the-art performance[34]. In principle, there are two means of handling cases that are not yet sufficiently covered by nnU-Net. For potentially recurring cases, nnU-Net's heuristics could be extended accordingly; for

highly domain-specific cases, nnU-Net should be seen as a good starting point for necessary modifications.

In summary, nnU-Net sets a new state of the art in various semantic segmentation challenges and displays strong generalization characteristics requiring neither expert knowledge nor compute resources beyond standard network training. As indicated by Litjens et al. and quantitatively confirmed here, method configuration in biomedical imaging used to be considered a "highly empirical exercise", for which "no clear recipe can be given" (ref. [10]). Based on the recipe proposed in this work, nnU-Net is able to automate this often insufficiently systematic and cumbersome procedure and may thus help alleviate this burden. We propose to leverage nnU-Net as an out-of-the box tool for state-of-the-art segmentation, as a standardized and dataset-agnostic baseline for comparison and as a framework for the large-scale evaluation of novel ideas without manual effort.

Online content

Any methods, additional references, Nature Research reporting summaries, source data, extended data, supplementary information, acknowledgements, peer review information; details of author contributions and competing interests; and statements of data and code availability are available at https://doi.org/10.1038/s41592-020-01008-z.

Received: 1 April 2020; Accepted: 29 October 2020;
Published online: 7 December 2020

References

1. Falk, T. et al. U-net: deep learning for cell counting, detection, and morphometry. *Nat. Methods* **16**, 67–70 (2019).
2. Hollon, T. C. et al. Near real-time intraoperative brain tumor diagnosis using stimulated Raman histology and deep neural networks. *Nat. Med.* **26**, 52–58 (2020).
3. Aerts, H. J. W. L. et al. Decoding tumour phenotype by noninvasive imaging using a quantitative radiomics approach. *Nat. Commun.* **5**, 4006 (2014).
4. Nestle, U. et al. Comparison of different methods for delineation of 18F-FDG PET-positive tissue for target volume definition in radiotherapy of patients with non-small cell lung cancer. *J. Nucl. Med.* **46**, 1342–1348 (2005).
5. De Fauw, J. et al. Clinically applicable deep learning for diagnosis and referral in retinal disease. *Nat. Med.* **24**, 1342–1350 (2018).
6. Bernard, O. et al. Deep learning techniques for automatic MRI cardiac multi-structures segmentation and diagnosis: is the problem solved? *IEEE Trans. Med. Imaging* **37**, 2514–2525 (2018).
7. Nikolov, S. et al. Deep learning to achieve clinically applicable segmentation of head and neck anatomy for radiotherapy. Preprint at https://arxiv.org/abs/1809.04430 (2018).
8. Kickingereder, P. et al. Automated quantitative tumour response assessment of MRI in neuro-oncology with artificial neural networks: a multicentre, retrospective study. *Lancet Oncol.* **20**, 728–740 (2019).
9. Maier-Hein, L. et al. Why rankings of biomedical image analysis competitions should be interpreted with care. *Nat. Commun.* **9**, 5217 (2018).
10. Litjens, G. et al. A survey on deep learning in medical image analysis. *Med. Image Anal.* **42**, 60–88 (2017).
11. LeCun, Y. 1.1 deep learning hardware: past, present, and future. In *2019 IEEE International Solid-State Circuits Conference* 12–19 (IEEE, 2019).
12. Hutter, F., Kotthoff, L. & Vanschoren, J. *Automated Machine Learning: Methods, Systems, Challenges*. (Springer Nature, 2019).
13. Bergstra, J. & Bengio, Y. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.* **13**, 281–305 (2012).
14. Simpson, A. L. et al. A large annotated medical image dataset for the development and evaluation of segmentation algorithms. Preprint at https://arxiv.org/abs/1902.09063 (2019).
15. Ronneberger, O., Fischer, P. & Brox, T. U-net: convolutional networks for biomedical image segmentation. In *MICCAI* (eds. Navab, N. et al) 234–241 (2015).
16. Landman, B. et al. MICCAI multi-atlas labeling beyond the cranial vault—workshop and challenge. https://doi.org/10.7303/syn3193805 (2015).
17. Litjens, G. et al. Evaluation of prostate segmentation algorithms for MRI: the PROMISE12 challenge. *Med. Image Anal.* **18**, 359–373 (2014).
18. Bilic, P. et al. The liver tumor segmentation benchmark (LiTS). Preprint at https://arxiv.org/abs/1901.04056 (2019).
19. Carass, A. et al. Longitudinal multiple sclerosis lesion segmentation: resource and challenge. *NeuroImage* **148**, 77–102 (2017).
20. Kavur, A. E. et al. CHAOS challenge—combined (CT–MR) healthy abdominal organ segmentation. Preprint at https://arxiv.org/abs/2001.06535 (2020).
21. Heller, N. et al. The KiTS19 challenge data: 300 kidney tumor cases with clinical context, CT semantic segmentations, and surgical outcomes. Preprint at https://arxiv.org/abs/1904.00445 (2019).
22. Lambert, Z., Petitjean, C., Dubray, B. & Ruan, S. SegTHOR: segmentation of thoracic organs at risk in CT images. Preprint at https://arxiv.org/abs/1912.05950 (2019).
23. Maška, M. et al. A benchmark for comparison of cell tracking algorithms. *Bioinformatics* **30**, 1609–1617 (2014).
24. Ulman, V. et al. An objective comparison of cell-tracking algorithms. *Nat. Methods* **14**, 1141–1152 (2017).
25. Heller, N. et al. The state of the art in kidney and kidney tumor segmentation in contrast-enhanced CT imaging: results of the KiTS19 challenge. In *Medical Image Analysis* vol. 67 (2021).
26. Çiçek, Ö., Abdulkadir, A., Lienkamp, S. S., Brox, T. & Ronneberger, O. 3D U-net: learning dense volumetric segmentation from sparse annotation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention* (eds. Ourselin, S. et al.) 424–432 (Springer, 2016).
27. Milletari, F., Navab, N. & Ahmadi, S.-A. V-net: fully convolutional neural networks for volumetric medical image segmentation. In *International Conference on 3D Vision (3DV)* 565–571 (IEEE, 2016).
28. He, K., Zhang, Z., Ren, S. & Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 770–778 (IEEE, 2016).
29. Jégou, S., Drozdzal, M., Vazquez, D., Romero, A. & Bengio, Y. The one hundred layers tiramisu: fully convolutional DenseNets for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops* 11–19 (IEEE, 2017).
30. Huang, G., Liu, Z., van der Maaten, L. & Weinberger, K. Q. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 4700–4708 (IEEE, 2017).
31. Oktay, O. et al. Attention U-net: learning where to look for the pancreas. Preprint at https://arxiv.org/abs/1804.03999 (2018).
32. Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K. & Yuille, A. DeepLab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE Trans. Pattern Anal. Mach. Intell.* **40**, 834–848 (2017).
33. McKinley, R., Meier, R. & Wiest, R. Ensembles of densely-connected CNNs with label-uncertainty for brain tumor segmentation. In *International MICCAI Brain Lesion Workshop* (eds. Crimi, A. et al.) 456–465 (Springer, 2018).
34. Heinrich, L., Funke, J., Pape, C., Nunez-Iglesias, J. & Saalfeld, S. Synaptic cleft segmentation in non-isotropic volume electron microscopy of the complete *Drosophila* brain. In *International Conference on Medical Image Computing and Computer-Assisted Intervention* (eds. Frangi, A.F. et al.) 317–325 (Springer, 2018).
35. Nolden, M. et al. The Medical Imaging Interaction Toolkit: challenges and advances. *Int. J. Comput. Assist. Radiol. Surg.* **8**, 607–620 (2013).
36. Castilla, C., Maška, M., Sorokin, D. V., Meijering, E. & Ortiz-de-Solórzano, C. 3-D quantification of filopodia in motile cancer cells. *IEEE Trans. Med. Imaging* **38**, 862–872 (2018).
37. Sorokin, D. V. et al. FiloGen: a model-based generator of synthetic 3-D time-lapse sequences of single motile cells with growing and branching filopodia. *IEEE Trans. Med. Imaging* **37**, 2630–2641 (2018).
38. Menze, B. H. et al. The Multimodal Brain Tumor Image Segmentation benchmark (BRATS). *IEEE Trans. Med. Imaging* **34**, 1993–2024 (2014).
39. Svoboda, D. & Ulman, V. MitoGen: a framework for generating 3D synthetic time-lapse sequences of cell populations in fluorescence microscopy. *IEEE Trans. Med. Imaging* **36**, 310–321 (2016).
40. Wu, Z., Shen, C. & van den Hengel, A. Bridging category-level and instance-level semantic image segmentation. Preprint at https://arxiv.org/abs/1605.06885 (2016).
41. He, K., Zhang, X., Ren, S. & Sun, J. Identity mappings in deep residual networks. In *European Conference on Computer Vision* (eds. Sebe, N. et al.) 630–645 (Springer, 2016).
42. Kingma, D. P. & Ba, J. Adam: a method for stochastic optimization. In *3rd International Conference on Learning Representations* (eds. Bengio, Y. & LeCun, Y.) (ICLR, 2015).
43. Ioffe, S. & Szegedy, C. Batch normalization: accelerating deep network training by reducing internal covariate shift. In *Proceedings of Machine Learning Research* Vol. 37 (eds. Francis Bach and David Blei) 448–456 (PMLR, 2015).
44. Ulyanov, D., Vedaldi, A. & Lempitsky, V. Instance normalization: the missing ingredient for fast stylization. Preprint at https://arxiv.org/abs/1607.08022 (2016).
45. Wiesenfarth, M. et al. Methods and open-source toolkit for analyzing and visualizing challenge results. Preprint at https://arxiv.org/abs/1910.05121 (2019).

Methods

A quick overview of the nnU-Net design principles can be found in Supplementary Note 2. This section provides detailed information on the implementation of these principles.

Dataset fingerprints. As a first processing step, nnU-Net crops the provided training cases to their non-zero region. While this had no effect on most datasets in our experiments, it reduced the image size of brain datasets, such as D1 (brain tumor) and D15 (multiple sclerosis lesion (MSLes)), substantially and thus improved computational efficiency. Based on the cropped training data, nnU-Net creates a dataset fingerprint that captures all relevant parameters and properties: image size (that is, number of voxels per spatial dimension) before and after cropping, image spacing (that is, the physical size of the voxels), modalities (read from metadata) and number of classes for all images, as well as the total number of training cases. Furthermore, the fingerprint includes the mean and s.d., as well as the 0.5 and 99.5 percentiles of the intensity values in the foreground regions, that is, the voxels belonging to any of the class labels, computed over all training cases.

Pipeline fingerprints. nnU-Net atomates the design of deep learning methods for biomedical image segmentation by generating a so-called pipeline fingerprint that contains all relevant information. Importantly, nnU-Net reduces the design choices to the very essential ones and automatically infers these choices using a set of heuristic rules. These rules condense domain knowledge and operate on the above-described data fingerprint and project-specific hardware constraints. These rule-based parameters are complemented by fixed parameters, which are data-independent, and empirical parameters, which are optimized during training.

Fixed parameters. *Architecture template.* All U-Net architectures configured by nnU-Net originate from the same template. This template closely follows the original U-Net[15] and its 3D counterpart[26]. According to our hypothesis that a well-configured plain U-Net is still difficult to surpass, none of our U-Net configurations made use of recently proposed architectural variations, such as residual connections[28,41], dense connections[29,30], attention mechanisms[31], squeeze-and-excitation networks[46] or dilated convolutions[32]. Minor changes with respect to the original architecture were made. To enable large patch sizes, the batch size of the networks in nnU-Net is small. In fact, most 3D U-Net configurations were trained with a batch size of only two (Fig. SN5.1a in Supplementary Note 5). Batch normalization[43], which is often used to speed up or stabilize training, does not perform well with small batch sizes[47,48]. We therefore used instance normalization[44] for all U-Net models. Furthermore, we replaced ReLU with leaky ReLUs[49] (negative slope, 0.01). Networks are trained with deep supervision; additional auxiliary losses are added in the decoder to all but the two lowest resolutions, allowing gradients to be injected deeper into the network and facilitating the training of all layers in the network. All U-Nets employ the very common configuration of two blocks per resolution step in both the encoder and decoder, with each block consisting of a convolution, followed by instance normalization and a leaky ReLU nonlinearity. Downsampling is implemented as strided convolution (motivated by representational bottleneck[50]) and upsampling is implemented as convolution transposed. As a trade-off between performance and memory consumption, the initial number of feature maps is set to 32 and doubled (halved) with each downsampling (upsampling) operation. To limit the final model size, the number of feature maps is additionally capped at 320 and 512 for 3D and 2D U-Nets, respectively.

Training schedule. Based on experience and as a trade-off between runtime and reward, all networks are trained for 1,000 epochs, with one epoch being defined as iteration over 250 mini-batches. Stochastic gradient descent with Nesterov momentum ($\mu = 0.99$) and an initial learning rate of 0.01 is used for learning network weights. The learning rate is decayed throughout the training following the 'poly' learning rate policy[32], $(1 - \text{epoch}/\text{epoch}_{\max})^{0.9}$. The loss function is the sum of cross-entropy and Dice loss[51]. For each deep supervision output, a corresponding downsampled ground truth segmentation mask is used for loss computation. The training objective is the sum of the losses (L) at all resolutions, $L = w_1 \times L_1 + w_2 \times L_2 + \ldots$ Hereby, the weights (w) are halved with each decrease in resolution, resulting in $w_2 = ½ \times w_1$, $w_3 = ¼ \times w_1$, etc. and are normalized to sum to 1. Samples for the mini-batches are chosen from random training cases. Oversampling is implemented to ensure robust handling of class imbalances; 66.7% of samples are from random locations within the selected training case, while 33.3% of patches are guaranteed to contain one of the foreground classes that are present in the selected training sample (randomly selected). The number of foreground patches is rounded with a forced minimum of 1 (resulting in one random and one foreground patch with a batch size of two). A variety of data augmentation techniques are applied on the fly during training: rotations, scaling, Gaussian noise, Gaussian blur, brightness, contrast, simulation of low resolution, gamma correction and mirroring. Details are provided in Supplementary Note 4.

Inference. Images are predicted with a sliding window approach, in which the window size equals the patch size used during training. Adjacent predictions overlap by half of the size of a patch. The accuracy of segmentation decreases toward the borders of the window. To suppress stitching artifacts and reduce the influence of positions close to borders, a Gaussian importance weighting is applied, increasing the weight of the center voxels in the softmax aggregation. Test time augmentation by mirroring along all axes is applied.

Rule-based parameters. *Intensity normalization.* There are two different image intensity normalization schemes supported by nnU-Net. The default setting for all modalities, except for CT images, is z-scoring. For this option, during training and inference, each image is normalized independently by first subtracting its mean and then dividing by its s.d. If cropping results in an average size decrease of 25% or more, a mask for central non-zero voxels is created, and normalization is applied within that mask only, ignoring the surrounding zero voxels. For CT images, nnU-Net employs a different scheme, as intensity values are quantitative and reflect physical properties of the tissue. It can therefore be beneficial to retain this information by using a global normalization scheme that is applied to all images. To this end, nnU-Net uses the 0.5 and 99.5 percentiles of the foreground voxels for clipping as well as the global foreground mean and s.d. for the normalization of all images.

Resampling. In some datasets, particularly in the medical domain, voxel spacing (the physical space that the voxels represent) is heterogeneous. Convolutional neural networks operate on voxel grids and ignore this information. To cope with this heterogeneity, nnU-Net resamples all images at the same target spacing (see paragraph below) using either third-order spline, linear or nearest-neighbor interpolation. The default setting for image data is third-order spline interpolation. For anisotropic images (maximum axis spacing ÷ minimum axis spacing > 3), in-plane resampling is performed with third-order spline interpolation, whereas out-of-plane interpolation is performed with nearest-neighbor interpolation. Treating the out-of-plane axis differently in anisotropic cases suppresses resampling artifacts, as large contour changes between slices are much more common. Segmentation maps are resampled by converting them to one-hot encodings. Each channel is then interpolated with linear interpolation, and the segmentation mask is retrieved by an argmax operation. Again, anisotropic cases are interpolated using 'nearest neighbor' on the low-resolution axis.

Target spacing. Selected target spacing is a crucial parameter. Larger spacings result in smaller images and thus a loss of detail, whereas smaller spacings result in larger images, preventing the network from accumulating sufficient contextual information, as the patch size is limited by the given graphics processing unit (GPU) memory budget. Although this trade-off is addressed in part by the 3D U-Net cascade (see below), a sensible target spacing for low and full resolution is still required. For the 3D full-resolution U-Net, nnU-Net uses the median value of the spacings found in the training cases computed independently for each axis as default target spacing. For anisotropic datasets, this default can result in severe interpolation artifacts or in a substantial loss of information due to large variances in resolution across the training data. Therefore, the target spacing of the lowest resolution axis is selected to be the tenth percentile of the spacings found in the training cases, if both voxel and spacing anisotropy (that is, the ratio of lowest spacing axis to highest spacing axis) are greater than three. For the 2D U-Net, nnU-Net generally operates on the two axes with the highest resolution. If all three axes are isotropic, the two trailing axes are used for slice extraction. The target spacing is the median spacing of the training cases (computed independently for each axis). For slice-based processing, no resampling along the out-of-plane axis is required.

Adaptation of network topology, patch size and batch size. Finding an appropriate U-Net architecture configuration is crucial for good segmentation performance. nnU-Net prioritizes large patch sizes while remaining within a predefined GPU memory budget. Larger patch sizes allow for more contextual information to be aggregated and thus typically increase segmentation performance. These come, however, at the cost of a decreased batch size, which results in noisier gradients during backpropagation. To improve the stability of the training, we require a minimum batch size of two and choose a large momentum term for network training (Fixed parameters). Image spacing is also considered in the adaptation process: downsampling operations may be configured to operate only on specific axes and convolutional kernels in the 3D U-Nets and may be configured to operate on certain image planes only (pseudo 2D). The network topology for all U-Net configurations is chosen on the basis of the median image size after resampling as well as the target spacing at which the images were resampled. A flow chart for the adaptation process is presented in Fig. SN5.1 in Supplementary Note 5. The adaptation of the architecture template, which is described in more detail in the following text, is computationally inexpensive. Because the GPU memory consumption estimate is based on feature map sizes, no GPU is required to run the adaptation process.

Initialization. The patch size is initialized as the median image shape after resampling. If the patch size is not divisible by 2^{n_d} ror each axis, where n_d is the number of downsampling operations, it is padded accordingly.

Architecture topology. The architecture is configured by determining the number of downsampling operations along each axis, depending on the patch size and voxel

spacing. Downsampling is performed until further downsampling would reduce the feature map size to less than four voxels, or the feature map spacings become anisotropic. The downsampling strategy is determined by the voxel spacing; high resolution axes are downsampled separately until their resolution is within a factor of two of the lower resolution axis. Subsequently, all axes are downsampled simultaneously. Downsampling is terminated for each axis individually, once the respective feature map constraint is triggered. The default kernel size for convolutions is $3\times3\times3$ and 3×3 for 3D U-Net and 2D U-Net, respectively. If there is an initial resolution discrepancy between axes (defined as a spacing ratio larger than two), the kernel size for the out-of-plane axis is set to one until the resolutions are within a factor of two. Note that the convolutional kernel size then remains at three for all axes.

Adaptation to GPU memory budget. The largest possible patch size during configuration is limited by the amount of GPU memory. Because the patch size is initialized to the median image shape after resampling, it is initially too large to fit into the GPU for most datasets. nnU-Net estimates the memory consumption of a given architecture based on the size of the feature maps in the network, comparing it to reference values of known memory consumption. The patch size is then reduced in an iterative process, while the architecture configuration is accordingly updated in each step until the required budget is reached (Fig. SN5.1 in Supplementary Note 5). The reduction of the patch size is always applied to the largest axis relative to the median image shape of the data. The reduction in one step amounts to $2n_d$ voxels of that axis, where n_d is the number of downsampling operations.

Batch size. As a final step, the batch size is configured. If a reduction of patch size was performed, the batch size is set to two. Otherwise, the remaining GPU memory headroom is used to increase the batch size until the GPU is fully used. To prevent overfitting, the batch size is capped, such that the total number of voxels in the mini-batch do not exceed 5% of the total number of voxels of all training cases. Examples for generated U-Net architectures are presented in sections 1 and 2 of Supplementary Note 3.

Configuration of the 3D U-Net cascade. Running a segmentation model on downsampled data increases the size of patches in relation to the image and thus enables the network to accumulate more contextual information. This comes at the cost of reduced detail in the generated segmentations and may also cause errors if the segmentation target is very small or characterized by its texture. In a hypothetical scenario with unlimited GPU memory, it is thus generally favored to train models at full resolution with a patch size that covers the entire image. The 3D U-Net cascade approximates this approach by first running a 3D U-Net on downsampled images and then training a second, full-resolution 3D U-Net to refine the segmentation maps of the former. In this way, the 'global', low-resolution network uses maximal contextual information to generate its segmentation output, which then serves as an additional input channel to guide the second, 'local' U-Net. The cascade is triggered only for datasets in which the patch size of the 3D full-resolution U-Net covers less than 12.5% of the median image shape. If this is the case, the target spacing for the downsampled data and the architecture of the associated 3D low-resolution U-Net are configured jointly in an iterative process. The target spacing is initialized as the target spacing of the full-resolution data. For the patch size to cover a large proportion of the input image, the target spacing is increased stepwise by 1% while updating the architecture configuration accordingly in each step, until the patch size of the resulting network topology surpasses 25% of the current median image shape. If the current spacing is anisotropic (the difference between lowest and highest resolution axes is a factor of two), only the spacing of the higher resolution axes is increased. The configuration of the second 3D U-Net of the cascade is identical to that of the standalone 3D U-Net, the configuration process of which is described above (except that the upsampled segmentation maps of the first U-Net are concatenated to its input). Fig. SN5.1b in Supplementary Note 5 provides an overview of this optimization process.

Empirical parameters. *Ensembling and selection of U-Net configuration(s).* nnU-Net automatically determines which (ensemble of) configuration(s) to use for inference, based on the average foreground Dice coefficient computed via cross-validation on the training data. The selected model(s) can be either a single U-Net (2D, 3D full-resolution, 3D low-resolution or the full-resolution U-Net of the cascade) or an ensemble of any two of these configurations. Models are ensembled by averaging softmax probabilities.

Post-processing. Connected component-based post-processing is commonly used in medical image segmentation[18,25]. Especially in organ image segmentation, it often helps to eliminate the detection of spurious false positives by removing all but the largest connected component. nnU-Net follows this assumption and automatically benchmarks the effect of suppressing smaller components on the cross-validation results. First, all foreground classes are treated as one component. If suppression of all but the largest region improves the average foreground Dice coefficient and does not reduce the Dice coefficient for any of the classes, this procedure is selected as the first post-processing step. Finally, nnU-Net builds on the outcome of this step and decides whether the same procedure should be performed for individual classes.

Implementation details. nnU-Net is implemented in Python 3.8.5 using PyTorch framework 1.6.0 (ref. [52]). Batchgenerators 0.21 (ref. [53]) is used for data augmentation. Python libraries that were also used include tqdm 4.48.2, dicom2nifti 2.2.10, scikit-image 0.17.2, MedPy 0.4.0, SciPy 1.5.2, batchgenerators 0.21, NumPy 1.19.1, scikit-learn 0.23.2, SimpleITK 1.2.4 and pandas 1.1.1. For use as a framework, nnU-Net's source code is available on GitHub (https://github.com/MIC-DKFZ/nnUNet). Users who wish to use nnU-Net as a standardized benchmark or to run inference with our pretrained models can install nnU-Net via PyPI. For a full description of how to use nnU-Net, as well as the most recent versions and dependencies, please refer to the online documentation available on the GitHub page.

Reporting Summary. Further information on research design is available in the Nature Research Reporting Summary linked to this article.

Data availability

All 23 datasets used in this study are publicly available and can be accessed via their respective challenge websites as follows. D1–D10 Medical Segmentation Decathlon, http://medicaldecathlon.com/; D11 Beyond the Cranial Vault (BCV)-Abdomen, https://www.synapse.org/#!Synapse:syn3193805/wiki/; D12 PROMISE12, https://promise12.grand-challenge.org/; D13 ACDC, https://acdc.creatis.insa-lyon.fr/; D14 LiTS, https://competitions.codalab.org/competitions/17094; D15 MSLes, https://smart-stats-tools.org/lesion-challenge; D16 CHAOS, https://chaos.grand-challenge.org/; D17 KiTS, https://kits19.grand-challenge.org/; D18 SegTHOR, https://competitions.codalab.org/competitions/21145; D19 CREMI, https://cremi.org/; D20–D23 Cell Tracking Challenge, http://celltrackingchallenge.net/.

Code availability

The nnU-Net repository is available as Supplementary Software. Updated versions can be found at https://github.com/mic-dkfz/nnunet. Pretrained models for all datasets used in this study are available for download at https://zenodo.org/record/3734294.

References

46. Hu, J., Shen, L. & Sun, G. Squeeze-and-excitation networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 7132–7141 (IEEE, 2018).
47. Wu, Y. & He, K. Group normalization. In *Proceedings of the European Conference on Computer Vision (ECCV)* (eds. Leal-Taixé, L. & Roth, S.) 3–19 (ECCV, 2018).
48. Singh, S. & Krishnan, S. Filter response normalization layer: eliminating batch dependence in the training of deep neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* 11237–11246 (CVPR, 2020)
49. Maas, A. L., Hannun, A. Y. & Ng, A. Y. Rectifier nonlinearities improve neural network acoustic models. In *Proceedings of the International Conference on Machine Learning* 3 (eds. Dasgupta, S. & McAllester, D.) (ICML, 2013).
50. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J. & Wojna, Z. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 2818–2826 (IEEE, 2016).
51. Drozdzal, M., Vorontsov, E., Chartrand, G., Kadoury, S. & Pal, C. The importance of skip connections in biomedical image segmentation. In *Deep Learning and Data Labeling for Medical Applications* (eds. Carneiro, G. et al.) 179–187 (Springer, 2016).
52. Paszke, A. et al. PyTorch: an imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems* (eds. Wallach, H. et al.) 8024–8035 (NeurIPS, 2019).
53. Isensee, F. et al. Batchgenerators—a Python framework for data augmentation. *Zenodo* https://doi.org/10.5281/zenodo.3632567 (2020).

Acknowledgements

This work was co-funded by the National Center for Tumor Diseases (NCT) in Heidelberg and the Helmholtz Imaging Platform (HIP). We thank our colleagues at DKFZ who were involved in the various challenge contributions, especially A. Klein, D. Zimmerer, J. Wasserthal, G. Koehler, T. Norajitra and S. Wirkert, who contributed to the Decathlon submission. We also thank the MITK team, which supported us in producing all medical dataset visualizations. We are also thankful to all the challenge organizers, who provided an important basis for our work. We want to especially mention N. Heller, who enabled the collection of all the details from the KiTS challenge through excellent challenge design, E. Kavur from the CHAOS team, who generated

comprehensive leaderboard information for us, C. Petitjean, who provided detailed leaderboard information of the SegTHOR entries from ISBI 2019 and M. Maška, who patiently supported us during our Cell Tracking Challenge submission. We thank M. Wiesenfarth for his helpful advice concerning the ranking of methods and the visualization of rankings. We further thank C. Pape and T. Wollman for their crucial introductions to the CREMI and Cell Tracking Challenges, respectively. Last but not least, we thank O. Ronneberger and L. Maier-Hein for their important feedback on this manuscript.

Author contributions

F.I. and P.F.J. conceptualized the method and planned the experiments with the help of S.A.A.K., J.P. and K.H.M.-H. F.I. implemented and configured nnU-Net and conducted the experiments on the 23 selected datasets. F.I. and P.F.J. analyzed the results and performed the KiTS analysis. P.F.J., S.A.A.K. and K.H.M.-H. conceived the communication and presentation of the method. P.F.J. designed and created the figures. P.F.J., F.I. and K.H.M.-H. wrote the paper with contributions from J.P. and S.A.A.K. K.H.M.-H. managed and coordinated the overall project. S.A.A.K. started work on this research as a PhD student at the German Cancer Research Center.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information is available for this paper at https://doi.org/10.1038/s41592-020-01008-z.

Correspondence and requests for materials should be addressed to K.H.M.-H.

Peer review information Rita Strack was the primary editor on this article and managed its editorial process and peer review in collaboration with the rest of the editorial team.

Reprints and permissions information is available at www.nature.com/reprints.

nature research

Corresponding author(s): Klaus Maier-Hein

Last updated by author(s): 15.10.2020

Reporting Summary

Nature Research wishes to improve the reproducibility of the work that we publish. This form provides structure for consistency and transparency in reporting. For further information on Nature Research policies, see Authors & Referees and the Editorial Policy Checklist.

Statistics

For all statistical analyses, confirm that the following items are present in the figure legend, table legend, main text, or Methods section.

n/a	Confirmed
☒	☐ The exact sample size (*n*) for each experimental group/condition, given as a discrete number and unit of measurement
☒	☐ A statement on whether measurements were taken from distinct samples or whether the same sample was measured repeatedly
☒	☐ The statistical test(s) used AND whether they are one- or two-sided *Only common tests should be described solely by name; describe more complex techniques in the Methods section.*
☒	☐ A description of all covariates tested
☒	☐ A description of any assumptions or corrections, such as tests of normality and adjustment for multiple comparisons
☒	☐ A full description of the statistical parameters including central tendency (e.g. means) or other basic estimates (e.g. regression coefficient) AND variation (e.g. standard deviation) or associated estimates of uncertainty (e.g. confidence intervals)
☒	☐ For null hypothesis testing, the test statistic (e.g. *F*, *t*, *r*) with confidence intervals, effect sizes, degrees of freedom and *P* value noted *Give P values as exact values whenever suitable.*
☒	☐ For Bayesian analysis, information on the choice of priors and Markov chain Monte Carlo settings
☒	☐ For hierarchical and complex designs, identification of the appropriate level for tests and full reporting of outcomes
☒	☐ Estimates of effect sizes (e.g. Cohen's *d*, Pearson's *r*), indicating how they were calculated

Our web collection on statistics for biologists contains articles on many of the points above.

Software and code

Policy information about availability of computer code

Data collection	No Software was used to collect the data for this study.
Data analysis	Our code is publicly available at github.com/mic-dkfz/nnunet. The following open source libraries have been utilized as part of our data analysis pipeline (all of them are python (python version 3.8.5) packages to be installed via "pip install <package_name>"): torch=1.6, tqdm=4.48.2, dicom2nifti=2.2.10, scikit-image=0.17.2, medpy=0.4.0, scipy=1.5.2, batchgenerators=0.21, numpy=1.19.1, scikit-learn=0.23.2, SimpleITK=1.2.4, pandas=1.1.1 .

For manuscripts utilizing custom algorithms or software that are central to the research but not yet described in published literature, software must be made available to editors/reviewers. We strongly encourage code deposition in a community repository (e.g. GitHub). See the Nature Research guidelines for submitting code & software for further information.

Data

Policy information about availability of data

All manuscripts must include a data availability statement. This statement should provide the following information, where applicable:

- Accession codes, unique identifiers, or web links for publicly available datasets
- A list of figures that have associated raw data
- A description of any restrictions on data availability

All 23 datasets used in this study are publicly available and can be accessed via their respective challenge website: D1-D10 Medical Segmentation Decathlon: http://medicaldecathlon.com/ ; D11 BCV-Abdomen: https://www.synapse.org/#!Synapse:syn3193805/wiki/ ; D12 PROMISE12: https://promise12.grand-challenge.org/ ; D13 ACDC: https://acdc.creatis.insa-lyon.fr/ ; D14 LiTS: https://competitions.codalab.org/competitions/17094 ; D15 MSLes: https://smart-stats-tools.org/lesion-challenge ; D16 CHAOS: https://chaos.grand-challenge.org/ ; D17 KiTS: https://kits19.grand-challenge.org/ ; D18 SegTHOR: https://competitions.codalab.org/competitions/21145 ; D19 CREMI: https://cremi.org/ ; D20-23 Cell Tracking Challenge: http://celltrackingchallenge.net/

Field-specific reporting

Please select the one below that is the best fit for your research. If you are not sure, read the appropriate sections before making your selection.

☒ Life sciences ☐ Behavioural & social sciences ☐ Ecological, evolutionary & environmental sciences

For a reference copy of the document with all sections, see nature.com/documents/nr-reporting-summary-flat.pdf

Life sciences study design

All studies must disclose on these points even when the disclosure is negative.

Sample size	We evaluate our algorithm on 23 publicly available datasets. We did not alter the sample size of utilized datasets, however, we experimentally confirm the importance of evaluation on a sufficient number of datasets and training cases in order to draw reliable conclusions.
Data exclusions	No data was excluded from the analysis.
Replication	All reported results are based on test sets from external challenges. Thus, scientific best practice dictates to perform only one submission (i.e. one replication) to avoid overfitting. During algorithm training we have not observed failed trainings or configurations, i.e. all attempted trainings were successful. The ablation studies in Figure 6 are reported as cross-validation results (1 run) from which 1000 bootstrap subsets were sampled to ensure ranking stability.
Randomization	All experiments are international segmentation competitions with pre-defined random splits between training data and test data.
Blinding	Experimenters were blinded with respect to the data collection process as well as the data splitting process and did not have access to labels of the corresponding test data.

Reporting for specific materials, systems and methods

We require information from authors about some types of materials, experimental systems and methods used in many studies. Here, indicate whether each material, system or method listed is relevant to your study. If you are not sure if a list item applies to your research, read the appropriate section before selecting a response.

Materials & experimental systems

n/a	Involved in the study
☒	☐ Antibodies
☒	☐ Eukaryotic cell lines
☒	☐ Palaeontology
☒	☐ Animals and other organisms
☒	☐ Human research participants
☒	☐ Clinical data

Methods

n/a	Involved in the study
☒	☐ ChIP-seq
☒	☐ Flow cytometry
☒	☐ MRI-based neuroimaging