Random Brains

Ulf Johansson, Tuve Löfström and Henrik Boström

***Abstract*—In this paper, we introduce and evaluate a novel method, called random brains, for producing neural network ensembles. The suggested method, which is heavily inspired by the random forest technique, produces diversity implicitly by using bootstrap training and randomized architectures. More specifically, for each base classifier multilayer perceptron, a number of randomly selected links between the input layer and the hidden layer are removed prior to training, thus resulting in potentially weaker but more diverse base classifiers. The experimental results on 20 UCI data sets show that random brains obtained significantly higher accuracy and AUC, compared to standard bagging of similar neural networks not utilizing randomized architectures. The analysis shows that the main reason for the increased ensemble performance is the ability to produce effective diversity, as indicated by the increase in the *difficulty* diversity measure.**

I. Introduction

WITHIN machine learning, it is well established that combining several individual classifiers into *ensembles* (a.k.a. *committee machines*, *mixtures of experts* and *multiple classifier systems*) will produce accurate and robust predictive models. Specifically, the use of ensemble models all but guarantees improved predictive performance, compared to single models, see e.g., [1] and [2]. An ensemble aggregates multiple classifiers (called *base classifiers*) into a composite model, making the ensemble prediction a function of all the included base classifiers.

The most intuitive explanation for why ensembles work is that the aggregation of several models, using averaging or majority voting, will eliminate uncorrelated base classifier errors; see e.g., [3]. Consequently, ensemble accuracy will be higher than mean base classifier accuracy, as long as the base classifiers commit their errors on different instances. Ideally, the base classifiers should make their mistakes independently. Informally, the key term *ensemble diversity* therefore describes how base classifier mistakes are distributed over the instances.

In [4], Brown et al. introduced a taxonomy of methods for creating diversity. The first obvious distinction made is between *explicit* methods, where some diversity metric is directly optimized, and *implicit* methods, where the method creates diversity without actually targeting it. Several implicit methods produce diversity by supplying each classifier with a slightly different training set. Standard *bagging* [5] obtains diversity by using resampling to create different training sets for each base classifier. More specifically, each training set (called a *bootstrap*) has the same size as the original training set, but since the instances are randomly selected with replacement, a bootstrap will contain multiple copies of some instances while lacking others completely. On average, approximately 63% of the orginal instances are present in each bootstrap. In the *random subspace method* [6], the base classifiers are trained on randomly chosen subspaces of the original attribute space, i.e., the individual training sets are instead sampled in the attribute (feature) space.

One important ensemble method producing diversity implicitly is *random forests* [7]. Standard random forests aggregate the base classifiers, a specific kind of decision trees called *random trees*, using majority voting. In random forests, the necessary diversity is introduced by combining the ideas from bagging and random subspacing. In more detail, each random tree is trained using a bootstrap, and only a randomly chosen subset of the attributes are used when calculating the best split in each node. Although the random forest method is quite straightforward, the resulting ensembles tend to be remarkably accurate. As a matter of fact, random forests often compare favorably to ensembles built using more elaborate schemes or consisting of more complex base classifiers, with regard to predictive performance.

In contrast to random forests, most ensemble techniques utilizing artificial neural networks (ANNs) as base classifiers have a tendency to be quite complicated, often explicitly optimizing some diversity metric. With this in mind, the overall purpose of this paper is to introduce and evaluate an implicit diversity method for producing ANN ensembles.

The suggested method, named *random brains*, is heavily inspired by the random forest technique with regard to how the necessary diversity is produced. More specifically, the overall idea of the random brains method is to employ training using bootstraps identically to random forest, while each ANN has a slightly randomized architecture between the input layer and the hidden layer, resulting in hidden units that only have access to a subset of the input attributes. The expected result of using randomized architectures is, of course, a further increase in diversity due to the fact that each hidden unit will be restricted to a randomized subset of the attributes when the weights are globally optimized during ANN training. Naturally, the inspiration for this way of introducing diversity is the procedure where each individual split in a random tree is optimized using only a randomized subset of the attributes.

In the experimentation, random brain ensembles are compared to ensembles of bagged ANNs; i.e., the difference is whether the randomized architectures are used or not. The

Ulf Johansson and Tuve Löfström are with the School of Business and IT at the University of Borås, Sweden. Henrik Boström is with the Department of Computer and Systems Sciences at Stockholm University, Sweden. email: {ulf.johansson,tuve.lofstrom}@hb.se, henrik.bostrom@dsv.su.se.

This work was supported by the Swedish Foundation for Strategic Research through the project High-Performance Data Mining for Drug Effect Detection (IIS11-0053) and the Knowledge Foundation through the project Big Data Analytics by Online Ensemble Learning (20120192).

analysis focuses on the predictive performance of the ensembles, using base classifier accuracy and different diversity measures to explain and discuss the results obtained.

II. Background.

Diversity measures are often divided into pairwise and non-pairwise measures. Pairwise measures calculate the average of a particular distance metric between all possible pairings of classifiers in the ensemble, while non-pairwise measures typically use some variation of *entropy*, or calculate a correlation of each ensemble member with the averaged output. In [8], Kuncheva and Whitaker summarize ten measures of classifier diversity: four pairwise and six non-pairwise measures. In this study, we will use two of the pairwise measures and one non-pairwise.

It must be noted that all diversity measures are in fact calculated on what is sometimes called an *oracle output matrix*, i.e., the correct target values are assumed to be known. Let the (oracle) output of each classifier D_i be represented as an N-dimensional binary vector y_i, where $y_{j,i} = 1$ if D_i correctly recognizes instance z_j and 0 otherwise. Let N^{ab} mean the number of instances for which $y_{j,i} = a$ and $y_{j,k} = b$. As an example, N^{11} is the number of instances correctly classified by both classifiers. N is the total number of instances.

The most intuitive diversity measure is probably the *disagreement* measure, which is the ratio between the number of instances on which one classifier is correct and the other incorrect to the total number of instances:

$$Dis_{i,k} = \frac{N^{10} + N^{01}}{N} \tag{1}$$

To find the diversity of a specific ensemble consisting of L classifiers, the averaged Dis over all pairs of classifiers is calculated:

$$Dis = \frac{2}{L(L-1)} \sum_{i=1}^{L-1} \sum_{k=i+1}^{L} Dis_{i,k} \tag{2}$$

For two-class problems, this measure simply calculates the proportion of instances where the two classifiers output different classes, i.e., it would not need oracle data but could be used directly on actual predictions. Naturally a higher disagreement value implies more diversity. The *double-fault* measure was proposed in [9] and is the proportion of instances misclassified by both classifiers:

$$DF_{i,k} = \frac{N^{00}}{N} \tag{3}$$

To calculate the double fault diversity for an ensemble, the double fault values are averaged over all pairs of classifiers, identically to disagreement. For double fault, a lower value indicates higher diversity.

The non-pairwise *difficulty* measure was introduced by Hansen and Salomon in [10]. Let X be a random variable taking values in $\{0/L, 1/L, \ldots, 1\}$. X is defined as the proportion of classifiers that correctly classify an instance x drawn randomly from the data set. To estimate X, all L classifiers are run on the data set. The difficulty is then defined as the variance of X. For difficulty, lower values mean higher diversity with the explanation that for a diverse classifier ensemble, even the hardest instances are often classified correctly by at least some base classifiers, resulting in a lower variance. The opposite would mean that all base classifiers are correct on some instances and wrong on some other instances, which of course would lead to higher variance.

Despite the fact that both intuition and a strong theoretical foundation advocate the benefit of diverse base classifiers, none of the suggested diversity measures is proven superior to the others. As a matter of fact, whether these diversity measures are useful as tools for ensemble optimization is questionable since solid empirical as well as theoretical validation of the explicit algorithms are absent from the field, [11]. When Kuncheva and Whitaker studied ten statistics measuring diversity using oracle outputs, i.e., correct or incorrect vote for the class label, all diversity measures evaluated showed low or very low correlation with test set accuracy, see [12]. In [13], Saitta supports Kuncheva's negative view, as presented in a series of papers, but she also goes one step further and shows not only that no useful measure exists today, but also that it is unlikely that one will ever exist. Clearly, these results seem to favor implicit methods. Consequently, it may be argued that the best use for diversity is not to use it during the optimization, but rather as a tool for explanation.

A. *Random forests.*

As described in the introduction, the random forest technique is a remarkably straightforward implicit method, often producing very accurate models. In addition, the random forest procedure has a number of nice properties. In the original paper, [7], Breiman lists five desirable characteristics:

1) Its accuracy is as good as Adaboost and sometimes better.
2) It it relatively robust to outliers and noise.
3) It is faster than bagging or boosting.
4) It gives useful internal estimates of error, strength, correlation and variable importance.
5) It is simple and easily parallelized.

In addition to this, another major advantage is the fact that the random forest procedure has very few parameter choices, and that the performance appears to be very robust with regard to different parameter values. When using an out-of-bag (OOB) estimator, the main idea is to get an unbiased estimator for test set accuracy by evaluating models trained using bagging on the instances that were not part of the bootstrap used for the actual training. Obviously, the main advantage, compared to setting aside a specific validation set, is that all available training data is actually used for building the ensemble. Producing an estimator for base classifier accuracy is straightforward, the OOB-estimate for each base classifier is simply the result of applying the trained base classifier to its OOB instances; i.e., the instances not used for training that specific base classifier. How to produce OOB-estimates for ensemble accuracy is slightly more complicated, but the standard method, as described by Breiman in the original paper, is to use different sub-ensembles on each instance. More specifically, for each instance, only the base classifiers that were not trained on that instance are allowed to vote. Since the sub-ensembles are much smaller than the original ensemble, OOB-estimates tend to underestimate the true ensemble test accuracy.

B. Related work.

Naturally, most standard methods for building ensembles have been applied to (and have often been modified for) ANN base classifiers. As an example, both standard bagging and boosting can be readily used on ANN classifiers. Comparing bagging and boosting, bagging has one inherent advantage since the models can be trained in parallel.

To produce more diverse ANN ensembles, Oza and Tumer [14] suggested using different subsets of the input features for each ANN, similar to the random subspace method. In [15], Raviv and Intrator proposed a method using bagging, weight decay and artificial noise to produce the diversity. Maclin and Shavlik [16] deliberately initialized the weights so different base classifiers would start out in different parts of the weight space. The DECORATE algorithm [17] creates diverse ANNs by producing different training sets for each base classifier by adding artificial training instances.

The negative correlation learning algorithm [18], trains ANNs simultaneously and interactively, trying to force different ANNs to learn different aspects of the data by introducing a correlation penalty term into the error function.

Finally, there are several evolutionary methods where diversity is a part of a fitness function and some evolutionary algorithm is used to search for an accurate ensemble. One example is the ADDEMUP method, suggested by Opitz and Shavlik in [19].

The most similar method to random brains is, however, when Cherkauer [20] produces diverse ANN base classifiers simply by using different number of hidden nodes. A very good survey of methods for producing diverse base classifiers, including ANNs, can be found in [4].

III. Method.

The overall idea of the random brains procedure is to mimic the basic properties of random forests in order to get a method that:

- produces diversity implicitly in order to build ANN ensembles that perform well in terms of predictive performance on most data sets.
- requires few parameter settings.
- is faster than other ANN ensemble schemes.
- makes it possible to utilize OOB estimations.
- is straightforward, making it easy to understand.
- is easily parallelized.

In this study, we introduce the basic random brains procedure, keeping it as simple as possible. With this in mind, the performance requirement is that a basic version must outperform standard bagging of similar ANNs.

In a random brain ensemble, each ANN is, as mentioned in the introduction, trained on a bootstrap. In addition, the architecture is, for each ANN, randomized in the following way:

1) The ANN starts out as a fully-connected multilayer perceptron (MLP) with one hidden layer. The number of input units is equal to the number of attributes. Since all data sets in this study are two-class problems, there is just one output unit. The number of hidden units is a parameter value, which must be determined for each data set. In the experimentation, a rule-of-thumb described below is used for this purpose.
2) For each link between the input layer and the hidden layer, a uniformly distributed random number in the interval $[0, 1]$ is generated. If the random number is smaller than another parameter value called *the cut*, that link is removed.

It must be noted that all links are removed *before* the training starts; i.e., the purpose of this step is just to randomize the architectures. Consequently, random brains is not another pruning procedure where the complexity of an already trained ANN is reduced in a controlled fashion in order to improve the generalization. For simplicity, we refer to the base classifiers in a random brain ensemble as *random nets*.

It is well-known that there is no general rule-of-thumb that will always find an optimal number of hidden units in an MLP, based on the data set characteristics. In practice, some kind of internal cross-validation is often used to determine the number of hidden units. Nevertheless, there are many rules-of-thumb proposed, and most of them suggest that the number of hidden units should be somewhere between the number of input units and the number of output units, thus resulting in pyramid shaped networks. In this study, we reluctantly decided to use a rule-of-thumb for determining the number of hidden units, in order to simplify the experimentation. More specifically, the number of hidden units in each network is $h = \lfloor \sqrt{\#attributes} \rfloor$. It should be noted that the use of this heuristics will result in smaller networks, compared to most other suggested rules-of-thumb. We have two main reasons for this choice:

1) Most heuristics are, at least implicitly, intended for single ANN models. Here, every ANN is supposed to be used as a base classifier in an ensemble, so maximizing their individual accuracies is not vital.
2) Since our suggested technique removes links from a fully-connected MLP, starting from overly complex ANNs could mean that pruning in itself would be beneficiary for the accuracy of individual models, thus resulting in an unfair advantage.

The data sets used are 20 two-class problems, publicly available from the UCI Repository [21]. Before the experimentation, each data set was preprocessed in the following way: first all missing numerical values were replaced with the mean value of that specific attribute, while missing categorical values were replaced with the mode values. Secondly, all categorical attributes were converted into binary attributes.

This study contains two experiments, described in detail below. All experimentation was carried out using MatLab version 2012b, including the neural network and statistics toolboxes.

A. Experiment 1.

The overall purpose of this experiment is to evaluate the effect of the randomized architectures in random brain ensembles. Altogether, five different setups with different levels of removed links are tried, using two different ensemble sizes. The setups are *bagg* (no removed links, i.e., bagged fully-connected MLPs), *rb20* (20% removed links), *rb40* (40% removed links), *rb60* (60% removed links) and *rb80* (80% removed links). For the actual evaluation, standard 10-fold cross-validation is used. In more detail: for each fold and setup, 150 ANNs are trained for 100 epochs using the resilient backpropagaton (rprop) learning algorithm. From this pool of trained ANNs, 50 random ensembles are formed of size 50, and 50 more random ensembles of size 100. The results reported on one data set for a specific setup and ensemble size are, consequently, averaged first over the random ensembles, and then over all ten folds.

The expected result is that base classifier accuracies will tend to be lower when more links are removed, while diversity should increase. The most interesting question is, however, how the ensembles will perform when built using the less powerful but more diverse random nets. Having said that, it is important to realize that the purpose of this first experiment is not to find a specific random brains variant that will outperform standard bagging, with regard to predictive performance, but to investigate and analyze the effect of using random nets in the ensembles.

In this experiment, we also evaluate the use of the OOB-estimator for selecting a specific setup. For each fold, we measure mean OOB base classifier accuracy and mean OOB ensemble accuracy for every setup, using either 50 or 100 ensembles, in order to select the best (according to these criteria) setup. The result of using the suggested setup (for that specific fold) is then compared to using standard bagging of fully-connected MLPs.

B. Experiment 2.

In the second experiment, the main purpose is to compare the predictive performance of several variations of the random brains method to standard bagging. More specifically, three increasingly sparse architectures are evaluated against bagging fully-connected MLPs. The four setups evaluated are described below:

- **bagg:** Standard bagging with fully-connected MLPs. The number of hidden units (h) is determined using the heuristics described above.
- **rb1.5x:** Bagged random nets with architectures where the number of hidden units is $1.5h$ and 33% of the links between the input layer and the hidden layer are removed.
- **rb2x:** Bagged random nets with architectures where the number of hidden units is $2h$ and 50% of the links between the input layer and the hidden layer are removed.
- **rb3x:** Bagged random nets with architectures where the number of hidden units is $3h$ and 66% of the links between the input layer and the hidden layer are removed.

It should be noted that using these setups, the number of links between the input layer and the hidden layer is constant over all four architectures. This experiment uses exactly the same evaluation procedure as Experiment 1, with the exception that only ensembles of size 100 are used.

This is of course the most important experiment, since the random brain idea is here directly compared to standard bagging. Although the sparse random nets have slightly more processing power in the form of more hidden units, it is not obvious how base classifier accuracies, diversity and, most importantly, ensemble accuracies are affected.

IV. Results.

In this section, the results from the two experiments are presented, analyzed and discussed.

A. Results Experiment 1.

Table I below shows the mean base classifier accuracies and disagreement for Experiment 1.

TABLE I
Experiment 1 - Base classifier accuracy and disagreement

	Base classifier accuracy					Disagreement				
Removed links	0%	20%	40%	60%	80%	0%	20%	40%	60%	80%
bcancer	.669	.666	.669	.670	.685	.259	.261	.257	.253	.237
breast-w	.940	.940	.940	.939	.930	.034	.032	.032	.033	.049
colic	.791	.789	.788	.788	.773	.167	.170	.172	.180	.211
credit-a	.839	.841	.841	.837	.817	.102	.103	.102	.111	.154
credit-g	.713	.714	.715	.719	.723	.239	.237	.232	.225	.217
diabetes	.758	.754	.744	.724	.695	.101	.115	.155	.199	.229
heart-c	.804	.805	.803	.804	.787	.158	.160	.161	.166	.195
heart-h	.800	.802	.805	.806	.797	.155	.153	.150	.152	.175
heart-s	.797	.797	.794	.790	.764	.158	.157	.166	.184	.235
hepatitis	.815	.813	.813	.810	.813	.156	.158	.162	.169	.176
iono	.872	.876	.878	.880	.875	.123	.123	.123	.126	.139
labor	.883	.880	.876	.874	.857	.097	.106	.114	.127	.171
liver	.661	.648	.629	.608	.587	.234	.268	.314	.333	.292
sick	.953	.954	.954	.953	.949	.023	.023	.025	.029	.033
sonar	.777	.781	.777	.776	.766	.214	.217	.217	.220	.235
spambase	.928	.928	.928	.926	.915	.044	.042	.042	.043	.066
spect	.781	.785	.785	.793	.797	.154	.151	.152	.151	.159
spectf	.785	.787	.789	.788	.781	.171	.169	.164	.166	.176
tic-tac-toe	.970	.965	.937	.869	.762	.032	.045	.096	.195	.265
vote	.946	.946	.943	.939	.918	.044	.046	.052	.062	.094
mean	**.824**	**.823**	**.820**	**.815**	**.799**	**.133**	**.137**	**.144**	**.156**	**.175**
mean rank	**2.53**	**2.25**	**2.65**	**3.30**	**4.28**	**3.75**	**3.68**	**3.53**	**2.55**	**1.50**

As described above, the disagreement metric specify how diverse the base classifiers are. One first, important, observation is that when comparing averaged base classifier accuracies over all data sets, the differences between the first three columns are fairly small, i.e., it is possible to remove up to 40% of the links without seriously reducing base classifier accuracy. Looking at the mean ranks, *rb-20* actually has the highest mean base classifier accuracy, i.e., even higher than when the base classifiers are fully-connected MLPs. To determine if there are any statistically significant differences, we use the statistical tests recommended by Demšar [22] for comparing several classifiers over a number of data sets; i.e., a Friedman test [23], followed by a Nemenyi post-hoc test [24]. With five setups and 20 data sets, the critical distance (for $\alpha = 0.05$) is as high as 1.36, so based on these tests the only significant differences are that *bagg*, *rb20* and *rb40* all produced more accurate base classifiers than *rb80*.

From the disagreement results it is obvious that the procedure of removing random links works as intended; i.e., more diverse ANNs are produced. Looking at both the averaged results over all data sets, and the mean ranks, the general picture is that the more links that are removed, the more diversity is introduced. In fact, the mean ranks identify three groups, ANNs produced by the *rb80* setup are clearly more diverse than ANNs produced by *rb60*, while the diversity produced by *rb60* is substantially higher than for *bagg*, *rb20* and *rb40*. Despite this clear ordering, it must be noted that the only statistically significant result is that the disagreement was significantly higher for *rb80* than *bagg*, *rb20* and *rb40*.

Turning to ensemble results, Table II below shows ensemble accuracies. Looking first at the mean results, i.e., ensemble accuracies averaged over all data sets, the differences are very small between the first four setups. So, the main result is that the diversity produced by removing randomly selected links managed to make up for the lower mean base classifier accuracies.

TABLE II
EXPERIMENT 1 - ENSEMBLE ACCURACY

ens. size	50 ANNs					100 ANNs				
setup	bagg	rb20	rb40	rb60	rb80	bagg	rb20	rb40	rb60	rb80
bcancer	.725	.734	.724	.712	.721	.729	.736	.725	.712	.720
breast-w	.947	.948	.946	.948	.951	.948	.948	.946	.948	.951
colic	.840	.839	.841	.849	.839	.841	.842	.845	.853	.836
credit-a	.851	.857	.859	.862	.862	.851	.856	.857	.862	.864
credit-g	.754	.748	.748	.754	.756	.753	.746	.747	.751	.755
diabetes	.767	.765	.766	.761	.729	.768	.764	.765	.761	.730
heart-c	.847	.844	.840	.845	.838	.853	.845	.844	.845	.841
heart-h	.828	.830	.833	.842	.843	.825	.824	.835	.843	.845
heart-s	.831	.834	.835	.839	.841	.827	.834	.834	.836	.840
hepatitis	.865	.864	.861	.853	.864	.865	.867	.864	.852	.866
iono	.910	.913	.915	.921	.927	.910	.913	.913	.920	.928
labor	.931	.933	.932	.931	.923	.933	.933	.933	.933	.924
liver	.703	.692	.684	.646	.594	.707	.695	.681	.646	.591
sick	.959	.959	.960	.959	.950	.958	.959	.961	.959	.951
sonar	.835	.837	.827	.834	.828	.835	.842	.832	.838	.825
spambase	.937	.937	.937	.936	.936	.937	.937	.937	.936	.936
spect	.809	.814	.822	.819	.830	.811	.815	.824	.821	.832
spectf	.810	.815	.814	.812	.804	.810	.817	.817	.812	.808
tic-tac-toe	.983	.984	.985	.982	.837	.983	.983	.985	.984	.839
vote	.958	.960	.960	.965	.959	.958	.960	.962	.965	.962
mean	**.855**	**.855**	**.854**	**.853**	**.842**	**.855**	**.856**	**.855**	**.854**	**.842**
mean rank	**3.13**	**2.80**	**2.85**	**2.90**	**3.33**	**3.23**	**2.80**	**2.90**	**2.88**	**3.20**

This result is even more apparent when comparing mean ranks. As a matter of fact, all setups have mean ranks between 3.33 and 2.80, so when considering all data sets evaluated, there are only marginal differences. As a consequence, we could reduce the training times for ANN base classifiers, simply by using networks with fewer links, without risking worse ensemble performance. Looking at the ranking ability, the AUC results in Table III below tell us that using fully-connected MLPs is actually the worst option.

TABLE III
EXPERIMENT 1 - ENSEMBLE AUC

ens. size	50 ANNs					100 ANNs				
setup	bagg	rb20	rb40	rb60	rb80	bagg	rb20	rb40	rb60	rb80
bcancer	.675	.666	.672	.672	.690	.676	.667	.674	.675	.692
breast-w	.988	.987	.986	.979	.961	.989	.988	.989	.986	.967
colic	.883	.871	.874	.878	.874	.886	.874	.876	.880	.875
credit-a	.914	.914	.915	.916	.919	.914	.914	.915	.917	.921
credit-g	.782	.787	.782	.786	.789	.785	.790	.784	.789	.792
diabetes	.807	.810	.820	.812	.796	.812	.818	.825	.816	.800
heart-c	.908	.912	.910	.914	.912	.908	.913	.912	.916	.913
heart-h	.900	.898	.904	.905	.906	.901	.898	.906	.906	.907
heart-s	.896	.895	.897	.905	.906	.897	.898	.899	.908	.906
hepatitis	.859	.863	.859	.862	.875	.866	.868	.863	.866	.879
iono	.974	.979	.977	.981	.979	.976	.981	.979	.983	.981
labor	.938	.944	.943	.950	.974	.944	.948	.950	.952	.975
liver	.729	.729	.728	.717	.684	.730	.732	.730	.725	.691
sick	.873	.875	.891	.899	.919	.885	.885	.902	.908	.927
sonar	.914	.923	.917	.917	.914	.916	.924	.920	.919	.917
spambase	.974	.974	.972	.970	.970	.976	.977	.974	.972	.972
spect	.782	.788	.791	.813	.829	.786	.792	.797	.816	.830
spectf	.878	.877	.877	.882	.876	.882	.882	.880	.886	.881
tic-tac-toe	.997	.998	.998	.995	.940	.998	.998	.998	.996	.945
vote	.988	.990	.988	.991	.990	.990	.990	.989	.992	.991
mean	**.883**	**.884**	**.885**	**.887**	**.885**	**.886**	**.887**	**.888**	**.890**	**.888**
mean rank	**3.60**	**3.05**	**3.30**	**2.45**	**2.60**	**3.55**	**3.00**	**3.30**	**2.45**	**2.70**

Even if the averaged AUC results are quite similar, the mean ranks show that *rb80* and *rb60* ensembles have clearly (but not significantly) higher AUCs, compared to standard bagging. If we further analyze individual data sets, some results are exactly what could have been expected. As an example, Fig. 1 below plots the results for the *diabetes* data set.

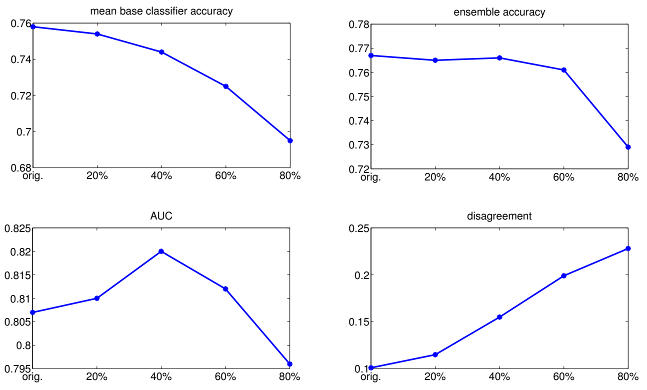

Fig. 1. Results diabetes data set

As seen in the top-left subplot, the mean base classifier accuracy indeed drops when the level of removed links is increased. Despite this, ensemble accuracies (top-right) and AUCs (bottom-left) are fairly constant, for all levels except 80%. Looking at the disagreement (bottom-right), the diversity increases, as expected, when the level of removed links is increased. For this data set, the general picture is thus that the obtained diversity manages to balance the loss of base classifier accuracy, until the base models are too weak. This is actually a quite common pattern, data sets with this behavior are *Colic*, *Diabetes*, *Labor*, *Sick*, *Sonar* and *Tic-Tac-Toe*. The most frequent pattern, however, is when the removed links do not result in decreased base classifier accuracy but stable or increasing base classifier accuracies, at least up to *rb80*. This pattern results in stable or often increasing predictive performances for the ensembles; see the results for *credit-a* in Fig. 2 below.

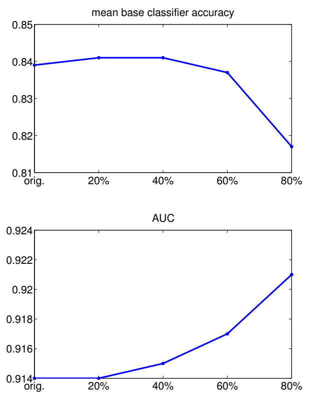

Fig. 2. Results credit-a data set

Here, when base classifier accuracy falls, disagreement instead increases rapidly, actually resulting in the most accurate setup being *rb80*. Data sets with similar results include *breast-w*, *credit-a*, *heart-h*, *heart-s*, *ionosphere*, *spambase* and *vote*. In fact, *liver-disorders*, see Fig. 3 below, is the only data set where decreasing base classifier accuracies lead to worsen ensemble performance.

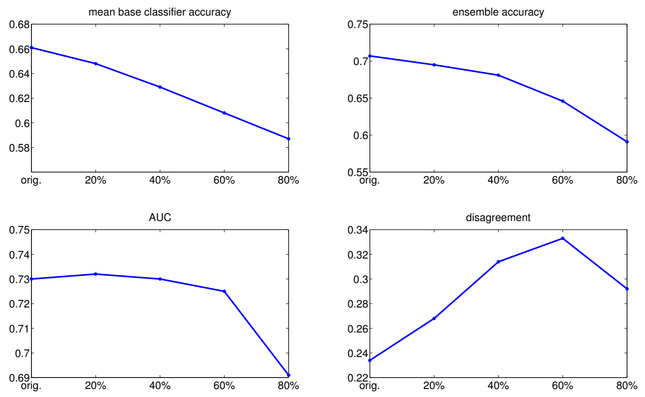

Fig. 3. Results liver disorders data set

From the results presented above, it is obvious that although the different setups all produced ensembles with similar predictive performance when averaged over all data sets, the differences for specific data sets are often not negligible. At the same time, there is no clear pattern identifying which setup that should be used for a specific data set.

Table IV below shows the result of using the OOB-estimator for selecting a specific setup. As described in the method section, this selection is performed on each fold. Comparing the use of the OOB-estimator to the five setups, the mean ranks show that it is at least better to use the ensemble accuracy OOB-estimator than just picking a setup at random. (With six setups compared, a random pick would of course get a mean rank of 3.5.) Just looking for accurate base classifiers, by picking the setup producing the highest OOB mean base classifier accuracy, on the other hand was not better than picking a random setup. Comparing, finally, the use of the OOB-estimator against standard bagging by counting wins, losses and ties, standard sign tests indicate that using the OOB-estimator to determine the setup is clearly better than just using standard bagging. As a matter of fact, using the ensemble accuracy OOB-estimator was, for the ensemble size of 100 ANNs, significantly better than standard bagging.

TABLE IV
EXPERIMENT 1 - USING OOB-ESTIMATES TO SELECT LEVEL OF REMOVED LINKS

50 ANNs:	Mean rank	W/T/L	p-value
Mean bAcc OOB	3.4	10/5/5	0.1509
EnsAcc OOB	3.0	12/1/7	0.1796
100 ANNs:	Mean rank	W/T/L	p-value
Mean bAcc OOB	3.5	10/5/5	0.1509
EnsAcc OOB	3.0	**13/2/5**	**0.0481**

Summarizing Experiment 1, removing links will generally lower base classifier accuracy, but at the same time increase diversity, leading to no decrease in ensemble predictive performance. The results also show that although no specific setup clearly outperformed standard bagging on its own, the procedure of using an ensemble accuracy OOB-estimator to select a specific setup did result in significantly higher accuracies, compared to standard bagging, at least for the ensemble size 100.

B. *Results Experiment 2.*

Table V below compares the three different random brain setups to standard bagging, with regard to accuracy and AUC. The most striking result is that all three random brains setups outperform standard bagging on a large majority of all data sets. As a matter of fact, standard bagging obtained the worst accuracy on 15 of the 20 data sets, and the worst AUC on 11 data sets.

TABLE V
EXPERIMENT 2 - PREDICTIVE PERFORMANCE

	Accuracy				AUC			
setup	bagg	rb1.5x	rb2x	rb3x	bagg	rb1.5x	rb2x	rb3x
bcancer	.697	.700	.706	.701	.676	.659	.668	.665
breast-w	.946	.947	.948	.946	.988	.988	.989	.988
colic	.836	.842	.841	.843	.885	.892	.883	.885
credit-a	.859	.860	.860	.865	.909	.913	.917	.915
credit-g	.753	.764	.759	.761	.793	.794	.790	.794
diabetes	.769	.769	.771	.771	.810	.821	.823	.827
heart-c	.846	.858	.851	.859	.910	.913	.913	.915
heart-h	.830	.832	.829	.836	.898	.899	.900	.905
heart-s	.837	.830	.846	.848	.897	.901	.901	.906
hepatitis	.839	.833	.843	.855	.866	.871	.870	.873
iono	.903	.905	.910	.915	.975	.975	.976	.979
labor	.923	.927	.927	.927	.970	.961	.976	.974
liver	.700	.707	.704	.706	.736	.741	.746	.749
sick	.960	.960	.961	.962	.890	.899	.901	.905
sonar	.829	.834	.831	.831	.922	.926	.926	.926
spambase	.938	.939	.939	.939	.976	.977	.976	.975
spect	.810	.813	.811	.811	.796	.788	.796	.813
spectf	.815	.817	.817	.818	.891	.888	.891	.897
tic-tac-toe	.983	.985	.985	.986	.998	.999	.999	.999
vote	.954	.956	.958	.957	.991	.990	.991	.988
mean	**.851**	**.854**	**.855**	**.857**	**.889**	**.890**	**.892**	**.894**
mean rank	**3.78**	**2.38**	**2.33**	**1.53**	**3.20**	**2.90**	**2.13**	**1.78**

For the statistical testing, we use another Friedman test, but now, since we treat standard bagging as the control, followed by a Bonferroni-Dunn post-hoc test [25]. With four setups and 20 data sets, the critical distance (for $\alpha = 0.05$) is 0.98. So, based on these tests, all three random brain setups obtained significantly higher accuracy than standard bagging. Furthermore, the ensembles produced by *rb2x* and *rb3x* also had significantly higher AUC than standard bagging. Obviously this is a very good result for the suggested random brains method, since it conclusively shows that when using bagging, utilizing a more sparse architecture where the exact link structure is randomized for each base classifier, will result in more accurate ensembles.

To further analyze this result, we start by looking at the average results over all data sets for mean base classifier accuracy and disagreement in Table VI below.

TABLE VI
EXPERIMENT 2 - BASE CLASSIFIER ACCURACY AND DISAGREEMENT

	Mean base accuracy				Disagreement			
setup	bagg	rb1.5x	rb2x	rb3x	bagg	rb1.5x	rb2x	rb3x
mean	.825	.824	.825	.825	.132	.136	.137	.137
mean rank	2.93	2.95	2.28	1.85	2.60	2.10	2.65	2.65

Just as expected, there are minimal differences in base classifier accuracy, when comparing the averaged results over all data sets. Still, based on the mean ranks, the more sparse random nets, i.e., *rb3x* and *rb2x*, have higher accuracy than the more dense nets on a clear majority of the data sets. Interestingly enough, there is no such apparent pattern describing how diversity, measured as disagreement, is affected by the different setups. Specifically, in this experiment, randomized architectures do not appear to produce more diverse base classifiers than standard bagging. Naturally, this was a somewhat unexpected and even discouraging result. At the same time, however, it seemed unlikely that the significant differences in predictive performance for random brain ensembles were due to just the slightly higher levels of mean base classifier accuracy. With this in mind, we decided to also use the two more informed diversity metrics *difficulty* and *double fault* for the analysis. As seen in Table VII below, the picture now becomes very different. According to these diversity measures, ensembles produced by all rb-setups are clearly, or even significantly, more diverse than ensembles produced by standard bagging. Specifically *rb3x*-ensembles are, when measuring either *difficulty* or *double fault*, significantly more diverse than standard bagging.

TABLE VII
EXPERIMENT 2 - DIFFICULTY AND DOUBLE FAULT

	Difficulty				Double fault			
setup	bagg	rb1.5x	rb2x	rb3x	bagg	rb1.5x	rb2x	rb3x
mean	.072	.070	.070	.069	.109	.108	.107	.106
mean rank	3.20	2.18	2.38	2.25	3.35	2.55	2.33	1.78

So, the use of randomized architectures resulted in only marginally more diverse ensembles, according to the intuitive, but fairly blind, disagreement measure. However, when instead considering diversity metrics acknowledging the difficulty of specific instances, it is obvious that the random brain method succeeded in producing the necessary diversity. Naturally, the fact that the kind of diversity produced actually improves the ensemble performance is another very good result for the suggested approach. Finally, we tabulate Spearman's rank correlations, between diversity and accuracy measures, based on the results from Experiment 2 in Table VIII below. It must be noted that these correlations are obtained on test data, i.e., they can only be used to explain the observed results. Actually using one of the diversity measures as part of an optimization scheme when building ensembles is, of course, something completely different. If nothing else, that would require that ensembles with relatively higher diversity on training or validation data would keep this edge on test data, and there is virtually no support for this in the overabundance of existing ensemble studies.

TABLE VIII
EXPERIMENT 2 - SPEARMAN'S RANK CORRELATIONS

	Ensemble accuracy	Mean base classifier accuracy
Mean base accuracy	0.27	-
Disagreement	0.04	-0.65
Difficulty	0.37	-0.04
Double fault	0.50	0.38

Interestingly enough, both *difficulty* and *double fault* have higher correlations with the ensemble accuracy than the mean base classifier accuracy. *Disagreement*, on the other hand, has an almost non-existing correlation with ensemble accuracy. This, together with the accuracy results presented above, are strong indicators that the random brain method is able to produce effective diversity. Finally, it could also be noted that although the correlation between *double fault* and ensemble accuracy is partly explained by the correlation between *double fault* and mean base classifier accuracy, this is not the case for *difficulty*. With this in mind, *difficulty* appears to be the most interesting diversity metric, of the three considered in this study, for analyzing ensemble performance.

V. Conclusions.

We have in this paper introduced and evaluated a novel method for producing ANN ensembles. The suggested method, which is heavily inspired by the random forest technique, combines bootstrap training with randomized architectures to produce the necessary diversity. From the experimentation, the overall picture is that using random nets tends to lower base classifier accuracy, compared to fully-connected MLPs, but at the same time diversity is increased, often leading to improved ensemble predictive performance. In a direct comparison with bagging of similar ANNs without the randomized architectures, all evaluated random brain setups obtained significantly higher accuracy. In addition, two setups of the three evaluated also had significantly higher AUC than standard bagging. This conclusively shows that adding randomized architectures to bagging will result in more accurate ANN ensembles.

When analyzing the results in terms of base classifier accuracy and diversity, the most interesting observation was that the use of randomized architectures did not result in significantly higher disagreement among the base classifiers, compared to fully-connected ANNs of similar size. When, however, using the more informed *difficulty* diversity measure, random brain ensembles were indeed significantly more diverse. This, together with the significantly better predictive performance, is a very strong indicator that the diversity produced by the method is effective for increasing ensemble accuracy.

Summarizing the properties of the suggested method, we find it to be straightforward and dependent on only a few parameter choices, making it easy to understand and analyze. Since it targets diversity implicitly and is inherently parallelizable, is will be much faster than most other ANN-ensemble schemes, especially those requiring interaction between the base classifiers during training. Finally, because bootstrap training is used, it allows the utilization of out-of-bag estimations.

VI. Discussion and future work.

It must be noted that the purpose of this paper has been to introduce random brains in a basic form. With this ambition, it becomes natural to only compare the performance to the most similar standard method, i.e., bagging. In future studies, however, random brains must be optimized, in order to be able to compete with more elaborate existing solutions. Some preliminary results comparing random brains to random forests show that the random nets were generally more accurate, but also much less diverse, than the corresponding random trees, leading to similar accuracies for the two techniques. From this, we believe that the (effective) diversity often must be further increased, even if it results in less accurate base classifiers, in order to find a better balance between base classifier accuracy and diversity. Fortunately, the random brains technique makes it easy to trade base classifier accuracy for diversity, in a controlled way.

References

[1] T. G. Dietterich, “Ensemble methods in machine learning,” in *Multiple Classifier Systems*, ser. Lecture Notes in Computer Science, J. Kittler and F. Roli, Eds., vol. 1857. Springer, 2000, pp. 1–15.

[2] D. W. Opitz and R. Maclin, “Popular ensemble methods: An empirical study,” *J. Artif. Intell. Res. (JAIR)*, vol. 11, pp. 169–198, 1999.

[3] T. G. Dietterich, “Machine-learning research: Four current directions,” *The AI Magazine*, vol. 18, no. 4, pp. 97–136, 1998.

[4] G. Brown, J. Wyatt, R. Harris, and X. Yao, “Diversity creation methods: a survey and categorisation,” *Journal of Information Fusion*, vol. 6, no. 1, pp. 5–20, 2005.

[5] L. Breiman, “Bagging predictors,” *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.

[6] T. K. Ho, “The random subspace method for constructing decision forests,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 8, pp. 832–844, 1998.

[7] L. Breiman, “Random forests,” *Machine Learning*, vol. 45, no. 1, pp. 5–32, October 2001.

[8] L. Kuncheva and C. Whitaker, “Ten measures of diversity in classifier ensembles: limits for two classifiers,” in *Intelligent Sensor Processing*, 2001, pp. 1001 – 1010.

[9] G. Giacinto and F. Roli, “Design of effective neural network ensembles for image classification purposes,” *Image and Vision Computing*, vol. 19, no. 9-10, pp. 699–707, 2001.

[10] L. K. Hansen and P. Salamon, “Neural network ensembles,” *IEEE Transactions on Pattern Analysis and Machine*, vol. 12, no. 10, pp. 993–1001, 1990.

[11] E. Tang, P. Suganthan, and X. Yao, “An Analysis of Diversity Measures,” *Machine Learning*, vol. vol 65, pp. 247–271, 2006.

[12] L. I. Kuncheva and C. Whitaker, “Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy,” *Machine Learning*, vol. 51, no. 2, pp. 181–207, 2003.

[13] L. Saitta, “Hypothesis Diversity in Ensemble Classification,” in *Foundations of Intelligent Systems*. Springer, 2006, pp. 662–670.

[14] N. C. Oza and K. Tumer, “Input decimation ensembles: Decorrelation through dimensionality reduction,” in *LNCS*. Springer, 2001, pp. 238–247.

[15] Y. Raviv and N. Intrator, “Bootstrapping with noise: An effective regularization technique,” *Connection Science*, vol. 8, pp. 355–372, 1996.

[16] R. Maclin and J. W. Shavlik, “Combining the predictions of multiple classifiers: Using competitive learning to initialize neural networks,” in *In Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*. Morgan Kaufmann, 1995, pp. 524–530.

[17] P. Melville and R. J. Mooney, “Creating diversity in ensembles using artificial data,” *Information Fusion*, vol. 6, pp. 99–111, 2004.

[18] Y. Liu and X. Yao, “Ensemble learning via negative correlation,” *Neural Networks*, vol. 12, pp. 1399–1404, 1999.

[19] D. W. Opitz, J. W. Shavlik, and O. Shavlik, “Actively searching for an effective neural-network ensemble,” *Connection Science*, vol. 8, pp. 337–353, 1996.

[20] K. Cherkauer, “Human expert-level performance on a scientific image analysis task by a system using combined artificial neural networks,” in *Working Notes of the AAAI Workshop on Integrating Multiple Learned Models*, 1996, pp. 15–21.

[21] A. Frank and A. Asuncion, “UCI machine learning repository,” 2010. [Online]. Available: http://archive.ics.uci.edu/ml

[22] J. Demšar, “Statistical comparisons of classifiers over multiple data sets,” *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, 2006.

[23] M. Friedman, “The use of ranks to avoid the assumption of normality implicit in the analysis of variance,” *Journal of American Statistical Association*, vol. 32, pp. 675–701, 1937.

[24] P. B. Nemenyi, *Distribution-free multiple comparisons. PhD-thesis*. Princeton University, 1963.

[25] O. J. Dunn, “Multiple Comparisons Among Means,” *Journal of the American Statistical Association*, vol. 56, no. 293, pp. 52–64, 1961.