Diagnosis of Melanoma Based on Data Mining and ABCD Formulas

Stanislaw BAJCAR

Regional Dermatology Center, 35-310 Rzeszow, Poland

Jerzy W. GRZYMALA-BUSSE

Department of Electrical Engineering and Computer Science, University of Kansas,
Lawrence, KS 66045, USA and
Institute of Computer Science, Polish Academy of Sciences,
01-237 Warsaw, Poland

Witold J. GRZYMALA-BUSSE

FilterLogix, Lawrence, KS 66049, USA

Zdzislaw S. HIPPE

Department of Expert Systems and Artificial Intelligence,
University of Information Technology and Management,
35-225 Rzeszow, Poland

Abstract. A parameter called TDS (Total Dermatoscopic Score), calculated by the well-known ABCD formula, is frequently used in melanoma diagnosis. In our previous research we found a new formula, similar to the original ABCD formula, that yielded fewer diagnostic errors. This new ABCD formula was developed using data mining techniques, in particular, the rule induction algorithm LEM2, a part of the data mining system LERS. In this paper we compare the quality of the old and new ABCD formulas, measured by the number of diagnostic errors, using three other data mining techniques: two rule induction algorithms, LEM1 and a modified version of LEM2 called MLEM2, and the decision tree generating system C4.5. Additionally, we compare the quality of diagnosis using TDS (original and new) and diagnosis without using TDS at all, to address complaints by some diagnosticians that TDS does not improve diagnosis of melanoma. Our experiments show that TDS is a valuable tool, significantly increasing melanoma diagnosis accuracy.

1. Introduction

In this paper we report on our recent results in improving diagnosis of melanoma based on data mining. Our data on melanoma were collected at the Regional Dermatology Center in Rzeszow, Poland [10]. The data consisted of 410 cases. In diagnosis of melanoma an important indicator is TDS, computed on the basis of the ABCD formula [3, 17], using four variables: *Asymmetry*, *Border, Color* and *Diversity*. The variable *Asymmetry* has three different values: *symmetric spot*, *one axial symmetry*, and *two axial symmetry*. *Border* is a numerical attribute, with values from 0 to 8. A lesion is partitioned into eight segments. The border of each segment is evaluated; the sharp border contributes 1 to *Border*, the gradual border contributes 0. *Color* has six possible values: *black*, *blue*, *dark brown*, *light brown*, *red* and *white*. Similarly, *Diversity* has five values: *pigment dots*, *pigment globules*, *pigment network*, *structureless areas* and *branched streaks*. In our data set *Color* and *Diversity* were replaced by binary single-valued variables. The TDS is traditionally computed using the following formula (known as the ABCD formula):

$$\text{TDS} = 1.3 * \text{Asymmetry} + 0.1 * \text{Border} + 0.5 * \Sigma\ \text{Colors} + 0.5 * \Sigma\ \text{Diversities},$$

where for *Asymmetry* the value *symmetric spot* counts as 0, *one axial symmetry* counts as 1, and *two axial symmetry* counts as 2, Σ Colors represents the sum of all values of the six color attributes and Σ Diversities represents the sum of all values of the five diversity attributes.

Most diagnosticians agree that TDS is a useful diagnostic tool for melanoma. However, there are some reports, for example [11], denying usefulness of TDS for melanoma diagnosis.

The obvious question is if the above ABCD formula can be improved, i.e., if there exists another formula, yielding less errors in diagnosis of melanoma. In our previous work [1, 7, 8, 9] we reported progress on finding an optimal ABCD formula.

Objectives of our current research, reported in this paper, were twofold:

- checking whether the optimal ABCD formula, found with the help of rough-set methodology, using LEM2 algorithm for rule induction, will remain optimal while changing data mining methodology, or, more specifically, with other rule induction algorithms, LEM1 and MLEM2 and with using production tree generation algorithm C4.5,
- checking whether TDS, original or optimal, improves diagnosis of melanoma.

2. Data Mining Methodology

Our basic data mining system was LERS (Learning from Examples based on Rough Sets). LERS may induce rules from inconsistent data, i.e., data with conflicting cases. Two cases are conflicting when they are characterized by the same values of all variables, but they belong to two different concepts. LERS handles inconsistencies using rough set theory, introduced by Z. Pawlak in 1982 [13, 14]. In rough set theory approach inconsistencies are not removed from consideration. Instead, lower and upper approximations of the concept are computed.

Let U denote the set of all cases (examples) of the data set and let P denote a nonempty subset of the set Q of all attributes. An *indiscernibility relation* ρ on U is defined for all $x, y \in U$ by $x \rho y$ if and only if for both x and y the values for all attributes from P are identical. Equivalence classes of ρ are called *elementary sets* of P. An equivalence class of ρ containing x is denoted $[x]_P$. Any finite union of elementary sets of P is called a *definable set* in P. Let X be a concept. In general, X is not a definable set in P. However, set X may be approximated by two definable sets in P, the first one is called a *lower approximation of* X *in* P and defined as follows

$$\{x \in U \mid [x]_P \subseteq X \}.$$

The second set is called an *upper approximation of* X *in* P and defined as follows

$$\{x \in U \mid [x]_P \cap X \neq \emptyset \}.$$

The lower approximation of X in A is the greatest definable set in A, contained in X. The upper approximation of X in A is the least definable set in A containing X. A *rough set* of X is the family of all subsets of U having the same lower and the same upper approximations of X.

Rules induced from the lower approximation of the concept *certainly* describe the concept, so they are called *certain*. On the other hand, rules induced from the upper approximation of the concept describe the concept only *possibly* (or *plausibly*), so they are called *possible. In* general, LERS uses two different approaches to rule induction: one is used in machine learning, the other in knowledge acquisition. In machine learning, or more specifically, in learning from examples, the usual task is to learn *discriminant description* [12], i.e., to learn the smallest set of minimal rules, describing the concept. To accomplish this goal, i.e., to learn discriminant description, LERS uses three algorithms: LEM1, LEM2, and MLEM2 (LEM1, LEM2, and MLEM2 stand for Learning from Examples Module, version 1 and 2, and Modified Learning from Examples Module 2, respectively). In our experiments rules were induced using all three the algorithms: LEM1 [4], LEM2 [4, 5] and MLEM2 [6].

The algorithm LEM1 computes minimal discriminant description of the concept on the basis of a single global covering. This algorithm employs the following ideas.

First new partitions on the set *U* of all cases are computed. Say that the original decision table described *k* concepts, i.e., decision has *k* values. Then 2*k* new partitions on *U*, called *substitutional partitions* [2] are created. Each substitutional partition has exactly two blocks, the first block is either lower or upper approximation of the concept, the second block is the complement of the first block. Substitutional partitions computed from lower approximations are called *lower substitutional partitions*; substitutional partitions computed from upper approximations are called *upper substitutional partitions*. Decisions, corresponding to lower and upper partitions, are called *lower* and *upper substitutional decisions* [2]. Let *d* denote a lower or upper substitutional decision. The family of all *P*-elementary sets will be denoted *P**. We say that {*d*} *depends on* P if and only if *P** ≤ {*d*}*. A *global covering* of {*d*} is a subset *P* of *Q* such that {*d*} depends on *P* and *P* is minimal in *Q*. The procedure LEM1 is presented below.

```
Procedure LEM1
input: the set Q of all attributes, partition {d}* on U;
output: a single global covering R;
begin
    compute partition Q*;
    P: = Q;
    R:= Ø;
    if Q* ≤ {d}*
        then
            begin
                for each attribute q in Q do
                    begin
                        S := P – {q};
                        compute partition S*;
                        if S* ≤ {d}* then P:= S
                    end {for}
                R:= P
            end {then}
end {procedure}.
```

On the basis of global covering certain and possible rules may be computed. However, further processing in the form of *dropping conditions* [12] is necessary. System LERS uses two different forms of dropping conditions. The first one is called *linear* because its time complexity is linear. For a rule of the form

$$C_1 \wedge C_2 \wedge \cdots \wedge C_l \rightarrow A,$$

linear dropping conditions means scanning the list of all conditions, from the left to the right, with an attempt to drop any of *l* condition, checking against the decision table where the simplified rule does not violate consistency of the discriminant description, where C_1, C_2,..., C_l are conditions and *A* is an action. Another possibility of LERS is *exponential dropping conditions*, in which any subset of the set of all *l* conditions of the above rule is checked for dropping conditions. Single global covering option of system LERS uses linear dropping conditions.

In algorithm LEM2 a rule set is induced by exploring the search space of blocks of attribute-value pairs. LEM2 induces a local covering and then converts it into the rule set. To define a local covering a few auxiliary definitions will be quoted. For a variable (attribute or decision) *x* and its value *v*, a block [(*x*, *v*)] of a variable-value pair (*x*, *v*) is the set of all cases for which variable *x* has value *v*.

Let *B* be a nonempty lower or upper approximation of a concept represented by a decision-value pair (*d*, *w*). Set *B depends* on a set *T* of attribute-value pairs (*a*, *v*) if and only if

$$\emptyset \neq [T] = \bigcap_{(a,\, v)\in T} [(a, v)] \subseteq B.$$

Set T is a *minimal complex* of B if and only if B depends on T and no proper subset T' of T exists such that B depends on T'. Let $\mathbb{T}$ be a nonempty collection of nonempty sets of attribute-value pairs. Then $\mathbb{T}$ is a *local covering of B* if and only if the following conditions are satisfied:

(1) each member T of $\mathbb{T}$ is a minimal complex of B,

(2) $\bigcup_{T\in\mathbb{T}} [T] = B$, and

(3) $\mathbb{T}$ is minimal, i.e., $\mathbb{T}$ has the smallest possible number of members.

The user may select an option of LEM2 with or without taking into account attribute priorities. The procedure LEM2 with attribute priorities is presented below. The option without taking into account priorities differs from the one presented below in the selection of a pair $t \in T(G)$ in the inner loop WHILE. When LEM2 is not to take attribute priorities into account, the first criterion is ignored. In our experiments all attribute priorities were equal to each other.

The procedure LEM2 is presented below.

```
Procedure LEM2
input: a set B,
output: a single local covering $\mathbb{T}$ of set B;
begin
    G := B;
    $\mathbb{T}$ := $\emptyset$;
    while G $\neq \emptyset$
        begin
        T := $\emptyset$;
        T(G) := {t | [t] $\cap$ G $\neq \emptyset$};
        while T = $\emptyset$ or [T] $\not\subseteq$ B
            begin
                    select a pair t $\in$ T(G) with the highest attribute
                    priority, if a tie occurs, select a pair t $\in$ T(G)
                    such that |[t] $\cap$ G| is maximum; if another tie
                    occurs, select a pair t $\in$ T(G) with the smallest
                    cardinality of [t]; if a further tie occurs, select
                    first pair;
                    T := T $\cup$ {t};
                    G := [t] $\cap$ G;
                    T(G) := {t | [t] $\cap$ G $\neq \emptyset$};
                    T(G) := T(G) – T;
            end {while}
        for each t in T do
            if [T – {t}] $\subseteq$ B then
                T := T – {t};
        $\mathbb{T}$ := $\mathbb{T}$ $\cup$ {T};
        G := B – $\bigcup_{T\in\mathbb{T}}$ [T];
    end {while};
    for each  T  in  $\mathbb{T}$  do
```

```
            if $\bigcup_{S \in \mathbb{T} - \{T\}} [S] = B$ then $\mathbb{T} := \mathbb{T} - \{T\}$;
end {procedure}.
```

For a set X, $|X|$ denotes the cardinality of X.

MLEM2 is a modified version of the algorithm LEM2. The original algorithm LEM2 needs discretization, a preprocessing, to deal with numerical attributes. Discretization is a process of converting numerical attributes into symbolic attributes, with intervals as values. LEM2 treats all attributes as symbolic, thus producing too specific rules when input data are not discretized. Also, the original LEM2 algorithm considers missing attribute values as special values.

In the algorithm MLEM2 discretization is performed simultaneously with rule induction. MLEM2 has an ability to recognize integer and real numbers as values of attributes, and labels such attributes as numerical. For numerical attributes MLEM2 computes blocks in a different way than for symbolic attributes. First, it sorts all values of a numerical attribute. Then it computes cutpoints as averages for any two consecutive values of the sorted list. For each cutpoint c MLEM2 creates two blocks, the first block contains all cases for which values of the numerical attribute are smaller than c, the second block contains remaining cases, i.e., all cases for which values of the numerical attribute are larger than c. The search space of MLEM2 is the set of all blocks computed this way, together with blocks defined by symbolic attributes. Starting from that point, rule induction in MLEM2 is conducted the same way as in LEM2.

The melanoma data set contained three numerical attributes: Asymmetry, Border and TDS. LERS in combination with LEM2 uses for discretization a number of discretization algorithms [1, 5, 15]. In our experiments were selected a polythetic divisive method of cluster analysis [15]. *Polythetic* methods use all attributes while *divisive* methods begin with all cases being placed in one cluster. Our method was also *hierarchical*, i.e., the final structure of all formed clusters was a tree.

Initially all cases were placed in one cluster C_1. Next, for every case the average distance from all other cases was computed. The case with the largest average distance was identified, removed from C_1, and placed in a new cluster C_2. For all remaining cases from C_1 a case c with the largest average distance d_1 from all other cases in C_1 was selected and the average distance d_2 from c to all cases in C_2 was computed. If $d_1 – d_2 > 0$, c was removed from C_1 and put to C_2. Then the next case c with the largest average distance in C_1 was chosen and the same procedure was repeated. The process was terminated when $d_1 – d_2 \leq 0$. The partition defined by C_1 and C_2 was checked whether all cases from C_1 were labeled by the same decision value and, similarly, if all cases from C_2 were labeled by the same decision value (though the label for C_1 might be different than the label for C_2). For clusters that contain cases labeled by at least two distinct decision values the same procedure of splitting into two clusters was repeated until all final clusters were labeled by the same decision value.

Once clusters were formed the postprocessing starts. First all clusters were projected on all attributes. Then the resulting intervals were merged to reduce the number of intervals and, at the same time, preserving consistency. Merging of intervals begins from *safe merging*, where, for each attribute, neighboring intervals labeled by the same decision value were replaced by their union. The next step of merging intervals was based on checking every pair of neighboring intervals whether their merging will result in preserving consistency. If so, intervals were merged permanently. If not, they were marked as un-mergeable. Obviously, the order in which pairs of intervals were selected affects the final outcome. In our experiments we processed attributes with the most intervals first.

The most important performance criterion for methods of data mining is the total number of errors. For evaluation of an error number we used the ten-fold cross validation: all cases were randomly re-ordered, and then the set of all cases was divided into ten mutually disjoint subsets of approximately equal size. For each subset, all remaining cases were used for training, i.e., for rule induction, while the subset was used for testing. Thus, each case was used nine times for training and once for testing. Note that using different re-orderings of cases cause slightly different error numbers. LERS may use *constant ten-fold cross validation* by using the same way of case re-ordering for all experiments. Also, LERS may perform ten-fold cross validation using different case re-orderings for every experiment, called *variable ten-fold cross validation*. On the other hand, some data mining systems, e.g., C4.5 [16], may use only constant n-fold cross validation.

3. Previous Experiments

Our previous experiments [1, 7, 9] were designed to find the optimal ABCD formula. We assumed that the optimal ABCD formula, for computing an optimal, new TDS, should be a linear combination of 13 attributes:

$$\begin{gathered}\text{new\_TDS} = c_1 * \text{Asymmetry} + c_2 * \text{Border} + c_3 * \text{Color\_black} + c_4 * \text{Color\_blue} \\ + c_5 * \text{Color\_dark\_brown} + c_6 * \text{Color\_light\_brown} + c_7 * \text{Color\_red} + c_8 * \text{Color\_white} \\ + c_9 * \text{Diversity\_pigment\_dots} + c_{10} * \text{Diversity\_pigment\_globules} \\ + c_{11} * \text{Diversity\_pigment\_network} + c_{12} * \text{Diversity\_structureless\_areas} \\ + c_{13} * \text{Diversity\_branched\_streaks.}\end{gathered}$$

Our objective was to find optimal values for coefficients $c_1, c_2,..., c_{13}$. The criterion of optimality was the smallest total number of errors for variable ten-fold cross validation for data with 13 old, unchanged attributes and with a new fourteenth attribute, new_TDS, that replaced the original TDS attribute.

In the first phase of experiments we searched through different vectors ($c_1, c_2,..., c_{13}$), using our discretization algorithm, based on divisive cluster analysis, and then using LEM2 rule induction algorithm. For each vector ($c_1, c_2,..., c_{13}$) the corresponding new_TDS was computed and then the sequence of five variable ten-fold cross validations was used for the evaluation of the number of errors. The smallest error indicated the optimal choice of ($c_1, c_2,..., c_{13}$).

A special script was created to compute the new_TDS given ranges for all 13 coefficients c_1, $c_2,..., c_{13}$. Experiments were conducted in sets of a few thousand at a time. Some overlapping occurred between such sets of experiments. In phase 1 the total number of executed variable ten-fold cross validations was about 40 thousand. Eventually, the following optimal ABCD formula was found [1]:

$$\begin{gathered}\text{new\_TDS} = 0.8 * \text{Asymmetry} + 0.11 * \text{Border} + 0.5 * \text{Color\_black} + 0.8 * \text{Color\_blue} \\ + 0.5 * \text{Color\_dark\_brown} + 0.6 * \text{Color\_light\_brown} + 0.5 * \text{Color\_red} \\ + 0.5 * \text{Color\_white} + 0.5 * \text{Diversity\_pigment\_dots} + 0.6 * \text{Diversity\_pigment\_globules} \\ + 0.5 * \text{Diversity\_pigment\_network} + 0.6 * \text{Diversity\_structureless\_areas} \\ + 0.6 * \text{Diversity\_branched\_streaks.}\end{gathered}$$

4. Current Experiments

A series of new experiments was performed to compare the original TDS attribute, called in the sequel old TDS, with a new TDS, computed by the above optimal ABCD formula. Additionally, a

Table 1. Average number of errors, results of 30 runs of variable ten-fold cross validation

	Old TDS	New TDS	No TDS
LEM1	19.30	19.60	62.50
LEM2	13.73	11.93	56.83
MLEM2	16.10	19.07	59.93

new data set was created, with the TDS attribute deleted. All three data sets, with old TDS, new TDS, and without TDS, were processed 30 times by variable ten-fold cross validation, using three different rule induction algorithms, LEM1, LEM2, and MLEM2. For experiments with LEM1 and LEM2, input data set was preprocessed using discretization based on divisive cluster analysis. Results of our experiments are presented in Tables 1 and 2.

In addition, all three data sets were preprocessed using a single run of the fixed ten-fold cross validation and three classifiers: two rule induction algorithms, LEM2 and MLEM2, and one decision tree induction algorithm C4.5. Results are presented in Table 3.

Table 2. Standard deviation, results of 30 runs of variable ten-fold cross validation

	Old TDS	New TDS	No TDS
LEM1	2.69	2.11	3.40
LEM2	1.39	1.53	3.78
MLEM2	2.89	2.77	3.24

Table 3. Number of errors, results of a single run of constant ten-fold cross validation

	Old TDS	New TDS	No TDS
LEM1	17	18	60
LEM2	15	10	48
MLEM2	17	17	57
C4.5	9	17	90

5. Conclusions

As follows from Tables 1 and 2, the new ABCD formula was found with the help of the LEM2 algorithm, hence, the new ABCD formula gives better results, at a 95% confidence level, when diagnosis is conducted with LEM2. On the other hand, for MLEM2, at the same confidence level, smaller number of errors were produced by data with the old TDS. For LEM1 both ABCD formulas, old and new, provide the same quality of diagnosis (the number of errors, at a 95% confidence level, is the same).

For a comparison of quality of two data mining systems a single run of constant ten-fold cross validation is not sufficient. The simplest statistical test is a difference of averages, when the distribution of the population is unknown. This test needs running variable n-cross validation for at least 30 times and requires knowledge of not only averages but also standard deviations. Table 3 shows that the number of errors for LEM1, LEM2, and MLEM2 for data with old TDS and new TDS values, as a result of a single running of constant ten-fold cross validation. For MLEM2 the number of errors for data with old and new TDS values is the same, in both cases the number of errors is 17, yet, as follows from running 30 variable ten-fold cross validation, the average number of errors is 16.10 for the old TDS and 19.07 for the new TDS (see Table 1). Taking into account standard deviations for both cases, it is not difficult to see that, with a 95% confidence level, data with old TDS values produce better results than data with new TDS values. Thus, on the basis of Table 3, we cannot conclude that running C4.5 and data with the old TDS will produce a decision tree with smaller number of errors than data with the new TDS, though it is highly likely. Among 30 experiments of variable ten-fold cross validation for LEM2 and old TDS, LEM2 also produced 9 errors, like C4.5.

Overall, the best classifier, at a 95% confidence level, is LEM2 with data using the new TDS.

Finally, our last observation is that TDS is a very helpful tool, definitely helping with diagnosis of melanoma. As follows from Table 1, diagnosis without taking TDS into account produces, at a

95% confidence level, more errors. On the basis of Table 3 we may conclude that it is highly probable that C4.5 produces many more errors when it is based on data without TDS. This last conclusion is surprising from an information theory view point since, in the data with TDS or without TDS, the information is the same (TDS is an additional attribute computed from remaining attributes, present in the data anyway).

Acknowledgment. This research has been partially supported by the State Committee for Research (KBN) of the Republic of Poland under the grant 7 T11E 030 21.

References

[1] A. Alvarez, S. Bajcar , F. M. Brown, J. W. Grzymala-Busse, and Z. S. Hippe, Optimization of the ABCD formula used for melanoma diagnosis. Proc. of the IIPWM'2003, International Conference on Intelligent Information Processing and WEB Mining Systems, Zakopane, Poland, 233–240.

[2] C. C. Chan and J. W. Grzymala-Busse, Rough-set boundaries as a tool for learning rules from examples. Proc. of the ISMIS–89, 4th Int. Symp. on Methodologies for Intelligent Systems, 281–288.

[3] R. J. Friedman, D. S. Rigel, and A. W. Kopf, Early detection of malignant melanoma: the role of physician examination and self-examination of the skin. *CA Cancer J. Clin.* **35** (1985) 130–151.

[4] J. W. Grzymala-Busse, LERS—A system for learning from examples based on rough sets. In: Intelligent Decision Support. Handbook of Applications and Advances of the Rough Sets Theory, ed. by R. Slowinski, Kluwer Academic Publishers, Dordrecht, Boston, 1992, 3–18.

[5] J. W. Grzymala-Busse, A new version of the rule induction system LERS. *Fundamenta Informaticae* **31** (1997) 27–39.

[6] J. W. Grzymala-Busse. MLEM2: A new algorithm for rule induction from imperfect data. Proc. of the IPMU 2002, 9th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, Annecy, France, 243–250.

[7] J. P. Grzymala-Busse, J. W. Grzymala-Busse, and Z. S. Hippe, Melanoma prediction using data mining system LERS. Proc. of the 25th Anniversary Annual International Computer Software and Applications Conference COMPSAC 2001, Chicago, IL.

[8] J. W. Grzymala-Busse and Z. S. Hippe, Postprocessing of rule sets induced from a melanoma data set. Proc. of the COMPSAC 2002, 26th Annual International Conference on Computer Software and Applications, Oxford, England.

[9] J. W. Grzymala-Busse and Z. S. Hippe, A search for the best data mining method to predict melanoma. Proceedings of the RSCTC 2002, Third International Conference on Rough Sets and Current Trends in Computing, Malvern, PA, Springer-Verlag, Berlin, Heidelberg, New York, 2002.

[10] Z. S. Hippe, Computer database NEVI on endargement by melanoma. *Task Quarterly* **4** (1999) 483–488.

[11] H. Lorentzen, K. Weismann, L. Secher, C. S. Peterson, and F. G. Larsen, The dermatoscopic ABCD rule does not improve diagnostic accuracy of malignant melanoma. *Acta Derm. Venereol.* **79** (1999) 469–472.

[12] R. S. Michalski, A theory and methodology of inductive learning. In: Machine Learning, R. S. Michalski, J. G. Carbonell, T. M. Mitchell (eds.), Morgan Kaufmann, Los Altos, CA, 1983, pp. 83–134.

[13] Z. Pawlak, Rough Sets. *International Journal of Computer and Information Sciences* **11** (1982) 341–356.

[14] Z. Pawlak, Rough Sets. Theoretical Aspects of Reasoning about Data. Kluwer Academic Publishers, Dordrecht, Boston, London, 1991.

[15] N. Peterson. Discretization using divisive cluster analysis and selected post-processing techniques. Department of Computer Science, University of Kansas, Internal Report, 1993.

[16] J. R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann, San Mateo, CA, 1993.

[17] W. Stolz, O. Braun-Falco, P. Bilek, A. B. Landthaler, and A. B. Cogneta, Color Atlas of Dermatology, Blackwell Science Inc., Cambridge, MA, 1993.